Universidad de Castilla-La Mancha

Escuela Superior de Ingeniería Informática

Departamento de Sistemas Informáticos

Programa Oficial de Postgrado en Tecnologías Informáticas Avanzadas

Trabajo Fin de Máster

Aprendizaje de reglas difusas lingüísticas cooperativas y heterogéneas usando búsqueda local: Extensión del espacio de búsqueda en COR.

Julio de 2011

Alumno: Javier Cózar del Olmo

Director: Dr. D. Luis de la Ossa Jiménez
Director: Dr. D. José Antonio Gámez Martín

Firma autor: Firma director/es:

---------- ----------

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

En esta memoria se presenta una descripción del trabajo desarrollado por Javier Cózar del Olmo durante el curso 2010/2011 en relación al Máster Universitario en Tecnologías Informáticas Avanzadas, el cual se engloba dentro del programa de postgrado en Tecnologías Informáticas Avanzadas con Verificación Oficial de la ANECA *Ref. MO2006-00197*, impartido por la Universidad de Castilla-La Mancha. La superación de este máster es un requisito necesario para la realización del programa de Doctorado en Tecnologías Informáticas Avanzadas, con Mención de Calidad de la ANECA Ref. MCD2006-00423, ofertado por la Universidad de Castilla-La Mancha.

Para completar este máster se requiere la aprobación de un total de 60 créditos ECTS, divididos básicamente en dos bloques. El primero consta de 30 créditos distribuidos entre 6 asignaturas (de las cuales 3 se han cursado el primer cuatrimestre y otras 3 el segundo). Los otros 30 créditos correspondientes al segundo bloque consisten en la realización de esta memoria (trabajo fin de master en Tecnologías Informáticas Avanzadas), la cual se estructura en los siguientes capítulos:

- **Capítulo 2:** En este capítulo se describirán las asignaturas cursadas en el máster, así como el porqué de su elección, el trabajo desarrollado para superarlas y cómo han afectado en la investigación del alumno.
- **Capítulo 3:** Aquí se expone el estado del arte relativo a la línea de investigación llevada a cabo durante este curso de máster, permitiendo contextualizar el problema a tratar y mostrando los principales métodos que existen en la actualidad para resolverlo.
- **Capítulo 4:** Se presenta la línea de investigación que se ha seguido en este trabajo y que se pretende continuar durante la realización del doctorado.
- **Capítulo 5:** Se muestran los objetivos alcanzados en la investigación realizada durante este curso.
- **Capítulo 6:** En este capítulo se presenta un resumen de las conclusiones obtenidas durante la realización de este máster.
- **Apéndice A:** Currículum vitae del alumno.

Capítulo 2

Asignaturas del máster

Las 6 asignaturas que son necesarias cursar y superar para lograr la obtención del título fueron seleccionadas entre un conjunto de 22 asignaturas (tanto en el primer cuatrimestre como en el segundo se eligen 3 asignaturas entre un conjunto de 11). Estas asignaturas son impartidas por personal docente de la Universidad de Castilla-La Mancha, distribuido entre los campus de Albacete y Ciudad Real. La decisión sobre qué asignaturas elegir estuvo basada en el campus en el que cada asignatura se impartía y en el contenido de las mismas, priorizando aquellas con una temática relacionada con la línea de investigación realizada.

Las asignaturas cursadas en el máster durante el primer cuatrimestre han sido *Sistemas inteligentes aplicados a Internet*, *Generación de documentos científicos en Informática* y *Nuevos paradigmas en HCI*. Respecto a las cursadas en el segundo cuatrimestre han sido *Programación en Internet con lenguajes declarativos multiparadigma*, *Computación en clusters* y *Modelos para el análisis y diseño de sistemas concurrentes*.

2.1. Sistemas inteligentes aplicados a Internet

Los profesores que han impartido esta asignatura durante el curso 2010/2011 han sido *María Julia Flores Gallego*, *José Miguel Puerta Callejón* e *Ismael García Varea*.

2.1.1. Contenidos

La impartición de esta asignatura se dividió en tres bloques bien diferenciados. En primer lugar, con la proferora *María Julia Flores Gallego* se trató el modelado y la inferencia en redes Bayesianas. En segundo lugar el profesor *José Miguel Puerta Callejón* procedió a explicar el aprendizaje de redes Bayesianas a partir de bases de datos y la recuperación de información sobre éstas últimas. Por último *Ismael García Varea* impartió conceptos sobre metaheurísticas y minería de datos.

Respecto a la primera parte, inicialmente se realizó una introducción a lo que son las redes Bayesianas y a los sistemas expertos probabilísticos. Posteriormente se procedió a explicar en más detalle el funcionamiento de una red Bayesiana, y las diferentes técnicas que se aplican para realizar inferencia sobre ellas.

En la segunda parte se incidió sobre el aprendizaje de estos sistemas, tanto su estructura como sus parámetros, mostrando diferentes métricas y procedimientos. También se trató sobre la problemática de disponer de grandes bases de datos con ejemplos incompletos, ya sea por registros dañados o por falta de información en las adquisiciones de los datos, y cómo tratar de solucionarlo proponiendo diferentes técnicas.

En la tercera parte en primer lugar se expusieron diferentes técnicas metaheurísticas para la resolución de determinados problemas, profundizando en los algoritmos genéticos y en los algoritmos de estimación de distribuciones (EDAs). Posteriormente se hizo una introducción a lo que es la minería de datos, y se expusieron las diferentes técnicas para clasificar los datos y evaluar estos modelos de clasificación.

2.1.2. Evaluación

Para superar esta asignatura fue necesario realizar tres trabajos, correspondientes a cada uno de los tres bloques en los que se dividió la asignatura. Estos trabajos constaban tanto de una parte teórica como de una parte práctica, de tal forma que se pudiese evaluar correctamente lo impartido en la asignatura. El primer trabajo consistía en una serie de ejercicios, de los cuales había que elegir como mínimo un subconjunto de ellos para realizar y entregar. Estos ejercicios cubrían los diferentes aspectos tratados en esta parte de la asignatura, tales como el modelado de redes Bayesianas y el proceso de inferencia sobre éstas. El segundo consistia en comprobar el funcionamiento de la métrica BIC realizando los cálculos con ayuda de una hoja de cálculo y corroborar que los resultados se ajustan a lo explicado en clase. También había que realizar un ejercicio de carácter teórico buscando información sobre un algoritmo de aprendizaje; y otro práctico, en el que se aplica uno de éstos sobre una base de datos dada. El tercer trabajo consistía en una relación de ejercicios, de los que se debía realizar un subconjunto de ellos. Éstos consistían en el diseño de un algoritmo genético para la resolución de un problema dado, obtener la estructura k-DB de una serie de variables discretas dada la información mútua entre ellas; y un ejercicio práctico en el que se debía utilizar la herramienta *Weka* para analizar la resolución de un problema dado mediante diferentes técnicas.

2.1.3. Conclusión

Sistemas inteligentes aplicados a Internet es una de las asignaturas que han sido de más importancia para mí, ya que en mi ámbito de investigación trato con diferentes técnicas metaheurísticas para la resolución de problemas combinatorios y aprendizaje de sistemas expertos basándome en bases de datos, por lo que el contenido de esta materia se ha ajustado muy bien a la temática de mi línea de investigación.

2.2. Generación de documentos científicos en Informática

Los profesores que han impartido esta asignatura durante el curso 2010/2011 han sido *José Antonio Gámez Martín*, *Francisco Parreño Torres* y *Luis de la Ossa Jiménez*.

2.2.1. Contenidos

La asignatura se dividió en tres bloques. El primero es impartido por *José Antonio Gámez Martín*. En el mismo se explica detalladamente en qué consiste el trabajo de un investigador, así como los diferentes rankings (dependiendo de la métrica que se utilice) en el que se puede medir la productividad investigacora. También se expuso la forma en la que hay que desarrollar documentos en este ámbito (las partes en las que se dividen, el formato que se les debe dar, ...), lo cual es útil tanto al redactar un trabajo como a la hora de corregirlos.

En el segundo bloque, impartido por *Francisco Parreño Torres*, se explicaron diferentes técnicas estadísticas para determinar si una hipótesis es falsa o no, y de esta forma, poder realizar análisis y comparaciones entre los resultados obtenidos en diferentes trabajos de investigación. Además de la base teórica sobre estas pruebas, se presentó un software de ámbito estadístico llamado *R* que sirve de apoyo para realizar estas pruebas.

En la última parte, impartida por *Luis de la Ossa Jiménez*, se presentó el sistema de composición de textos LaTeX como alternativa a otros procesadores de texto. Se explicó el funcionamiento de un entorno (*TeXnicCenter*) para generar documentos científicos utilizando este sistema, incidiendo en las opciones más utilizadas tales como edición de fórmulas matemáticas, elección del tipo de documento a generar y estructuración de su contenido. Además de esto, se mencionarion ciertos programas que ayudan a la automatización de ciertas tareas, como la generación de tablas y bibliografía.

2.2.2. Evaluación

Para superar esta asignatura fue necesario realizar dos trabajos. El primero, para aprobar la primera parte, consistia en aplicar diferentes métricas sobre el grupo de investigación en el que se esté trabajando (o uno cualquiera en su defecto) para comparar los resultados obtenidos entre los diferentes miembros del grupo y entre las diferentes métricas existentes.

El segundo consistía en hacer un análisis estadístico sobre un trabajo previamente realizado y obtener ciertas conclusiones sobre él. Para redactar el trabajo y realizar la presentacion correspondiente al mismo se debería utilizar LaTeX, quedando de esta manera cubiertos los bloques dos y tres. La presentación debería ser un resumen del trabajo desarrollado, centrándose en aquellos aspectos que son más importantes. La duración de la exposición no debía extenderse más de 15 minutos.

2.2.3. Conclusión

Considero esta asignatura como pilar base para iniciar correctamente una carrera investigadora, pues en ella se imparten los conocimientos básicos que se deben

conocer antes de realizar cualquier trabajo o investigación. Además el conocimiento sobre el correcto uso de LaTeX es de gran utilidad y permite ahorrar tiempo y mejorar la calidad de los documentos generados una vez que se ha adquirido la práctica necesaria.

2.3. Nuevos paradigmas en HCI

Los profesores que han impartido esta asignatura durante el curso 2010/2011 han sido *María Teresa López Bonal*, *José Pascual Molina Massó* y *Antonio Fernández Caballero*.

2.3.1. Contenidos

Al igual que las anteriores asignaturas, ésta también se estructuró en tres bloques. El primero lo impartió *José Pascual Molina Massó*, el segundo *María Teresa López Bonal* y el tercero *Antonio Fernández Caballero*.

En el primer bloque se presentó la problemática que existe en torno a las interfaces y qué diferentes soluciones se han propuesto para tratar de cubrirlos, analizando las ventajas e inconvenientes de cada una de ellas. Posteriormente se dio pie a un aspecto más práctico de la asignatura, como es el desarrollo de entornos tridimensionales y el manejo de dispositivos hardware para diseñar interfaces, utilizando para ello la librería OpenGL. También se utilizó una interfaz física (un casco de realidad virtual, unas cámaras y unos guantes con elementos guía) para interactuar con el entorno diseñado, y se implementó y analizó una técnica (*Virtual Hand*) para manipular objetos virtuales en este tipo de escenarios.

El segundo bloque se enfocó sobre la visión artificial, estudiando los procesos por los que pasa cada imagen para poder extraer información útil de ella. Una vez explicados estos conceptos teóricos se presentó la librería OpenCV, que implementa todas las funciones necesarias para llevar a cabo cada uno de estos procesos.

Por último, en el tercer bloque se presentó la forma en la que las empresas utilizan las técnicas impartidas en la asignatura y cómo y para qué utilizan las interfaces estudiadas en diferentes entornos.

2.3.2. Evaluación

Para superar la asignatura fue necesario asistir a clase y realizar los ejercicios prácticos solicitados en las mismas, así como la realización de un trabajo final de la asignatura y su correspondiente exposición. El trabajo debería estar relacionado con cualquiera de los temas que se han explicado en la asignatura y, a ser posible, que tengan relación con la línea de investigación que se esté realizando. En mi caso, el trabajo estuvo relacionado con el segundo bloque de la asignatura, utilizando OpenCV para generar diferentes salidas gráficas, mejorando la interfaz de un programa utilizado en mi línea de investigación. Además se analizó una librería gráfica (para el lenguaje Java) llamada *Processing* y una implementación de la librería OpenCV en este mismo lenguaje.

2.3.3. Conclusión

Inicialmente elegí esta asignatura porque su contenido me pareció interesante al tratar temas de las diferentes interfaces que se están presentando en el mercado, así como la forma en la que se pueden diseñar. Al cursarla he visto que, aunque no tenga demasiada relación con mi línea de investigación, simplemente la generación de imagenes y entornos virtuales tiene una gran aplicación en cualquier campo. Tanto es así que el trabajo que realicé para esta asignatura me ha servido para acoplarlo al trabajo de investigación desarrollado durante el curso para mejorar la estética de la información de salida generada por el software implementado.

2.4. Programación en Internet con lenguajes declarativos multiparadigma

Los profesores que han impartido esta asignatura durante el curso 2010/2011 han sido *Ginés Damián Moreno Valverde*, *Francisco Pascual Romero Chicharro* y *Pascual Julián Iranzo*.

2.4.1. Contenidos

La asignatura ha sido impartida a través de videoconferencia entre los campus de Ciudad Real y de Albacete. Cada profesor ha impartido un aspecto diferente de la asignatura. Primero *Pascual Julián Iranzo* procedió a explicar la base teórica y matemática de la lógica difusa, así como los pasos de unificación y resolución para extraer conocimiento a partir de la declaración de un conjunto de cláusulas. Para terminar con este bloque, se explicó la unificación débil y la resolución débil, así como su utilización en el lenguaje Bousi-Prolog.

Posteriormente el profesor *Ginés Damián Moreno Valverde* procedió a explicar la programación lógica difusa con grados de verdad, comenzando en primer lugar por el lenguaje PROLOG, y mostrando su evolución hasta MALP. También se vio la problemática referente al orden de complejidad de los programas así como la optimización de los mismos utilizando técnicas de *plegado-desplegado*.

Por último, el profesor *Francisco Pascual Romero Chicharro* impartió el bloque más práctico de la asignatura, aplicaciones de la programación declarativa a la búsqueda en internet, así como la forma en la que Bousi-Prolog puede ayudar a la resolución de ciertos problemas como es la construcción de directorios.

2.4.2. Evaluación

La asistencia era obligatoria para aprobar esta asignatura, ya que en clase con relativa frecuencia se proponían ejercicios para resolverlos. Además, para optar a una nota mayor que la de *aprobado* se podía realizar un trabajo final para exponerlo en las clases finales de la asignatura.

En mi caso decidí realizar este trabajo optativo sobre un lenguaje de programación llamado PARLOG. Éste es una variante del lenguaje PROLOG, capaz de llevar a cabo la ejecución de los programas de forma paralela. Es por tanto una variante orientada a la optimización de los tiempos de ejecución. La exposición

no debía sobrepasar los 15 minutos, e intentar que no fuese demasiado densa ya que las diferentes temáticas de los trabajos podían hacer que ciertos conceptos, explicados de forma muy detallada, quedasen fuera del alcance de comprensión de los alumnos.

2.4.3. Conclusión

En esta asignatura se puede apreciar como la programación declarativa tiene muchísimas ramas de aplicación. Observando las ventajas que proporciona junto con la reciente explotación en el uso de la temática difusa (y teniendo en cuenta la inclusión de esta vertiente en los lenguajes lógicos, llamados lógico-difusos) se puede comprender que esté siendo tan utilizada. En mi línea de investigación estoy tratando con lógica difusa (en concreto con sistemas basados en reglas difusas lingüísticas), por lo que lo impartido en esta asignatura me ha parecido muy útil e interesante.

2.5. Computación en clusters

Los profesores que han impartido esta asignatura durante el curso 2010/2011 han sido *José Luís Sánchez García*, *Francisco José Alfaro Cortes* y *Francisco José Quiles Flor*.

2.5.1. Contenidos

En esta asignatura se ha mostrado y justificado la evolución que han sufrido los computadores a lo largo de la historia, desde el computador con un monoprocesador hasta los clusters. *Francisco José Quiles Flor* fué el profesor que impartió el primero de los tres bloques de la asignatura. En éste se mostró las diferentes alternativas que existen a la hora de diseñar un cluster, cada una de éstas con sus correspondientes ventajas e inconvenientes.

En el segundo bloque *Francisco José Alfaro Cortes* se encargó de explicar en profundidad el funcionamiento de la E/S en sistemas de ficheros y la planificación de los trabajos en clusters. Para ello se intentó hacer ver a los alumnos la problemática existente, que si bien en sistemas monoprocesador no es un problema grave o ya sea ha solucionado en gran medida, cuando el sistema es tan grande como un cluster no es una tarea nada fácil.

Por último en el tercer bloque, *José Luís Sánchez García* mostró las diferentes vías que existen para programar estos sistemas paralelos, como puede ser utilizar MPI o CUDA. Además de programar utilizando estos sistemas a través del uso de ciertas librerías, también nos enfrentamos a la problemática de cómo paralelizar un código con dependencias entre datos (ya sea dependendias inter o intra iteración).

2.5.2. Evaluación

Para aprobar esta asignatura, además de ser necesario asistir presencialmente a un 80 % o más de las clases, es necesario realizar un trabajo final relacionado con la asignatura y, en la medida de lo posible, relacionado con nuestra línea de investigación. En mi caso me decanté por realizar un trabajo relacionado con el tercer bloque de la asignatura. El mismo consistió en la utilización simultánea de

CUDA y MPI para paralelizar un código que aplicaba la operación de la mediana sobre una imagen con una rejilla de tamaño 5×5 para reducir el ruido presente en la misma. Una vez realizada esta implementación se procedió a estudiar las ganancias que esta alternativa presenta, diseñando posteriormente una presentación en la que se exponía el trabajo realizado así como los resultados y conclusiones obtenidas.

2.5.3. Conclusión

En el ámbito de la investigación es muy difícil no haber oido hablar de un cluster, e incluso haberlo utilizado para ejecutar experimentos. Por tanto, esta asignatura es muy importante para conocer algunas de las técnicas existentes para paralelizar código secuencial. Además, existen ciertos estudios en los que se menciona la paralelización de algoritmos utilizados en minería de datos [14], lo cual esta relacionado estrechamente con mi línea de investigación.

2.6. Modelos para el análisis y diseño de sistemas concurrentes

Los profesores que han impartido esta asignatura durante el curso 2010/2011 han sido *Gregorio Díaz Descalzo*, *Valentín Valero Ruiz*, *María Emilia Cambronero Piqueras* y *Fernando Cuartero Gómez*.

2.6.1. Contenidos

Esta asignatura trata sobre la modelización de sistemas para la extracción de propiedades de los mismos. Dentro de las diferentes técnicas de modelización, *Valentín Valero Ruiz* se centró en las Redes de Petri, *Gregorio Díaz Descalzo* y *María Emilia Cambronero Piqueras* en las lógicas temporales LTL, CTL y TCTL y *Fernando Cuartero Gómez* en las Álgebras de Procesos.

Respecto a las Redes de Petri se profundizó tanto en el funcionamiento y uso de las mismas como también en el proceso matemático que se hizo para la obtención de sus propiedades y características. Se presentaron diferentes variantes de las Redes de Petri que inluyen características adicionales, mostrando a su vez problemas que sin estas ampliaciones no podrían ser modelados mediante este tipo de redes. También se vio un simulador de Redes de Petri que permite diseñar y verificar propiedades sobre éstas.

Respecto a las lógicas temporales se presentó su sintáxis y su semántica, incluyendo ejemplos para comprobar los resultados del modeladado y la verificación de ciertas propiedades sobre ellos. También se utilizó la herramienta UPPAAL y se realizaron ciertos ejercicios, modelando sistemas como una máquina de estados y comprobando propiedades sobre ellas con la sintáxis de TCTL.

Sobre las álgebras de procesos se vio su base teórica (sintáxis y semántica operacional) y sus ventajas e inconvenientes. También se explicó el porqué no se utiliza esta técnica para modelar sistemas, pues la explosión de estados lo hace inviable.

2.6.2. Evaluación

Para aprobar esta asignatura fue necesario la asistencia a clase, así como la realización de los ejercicios que se iban proponiendo en clase. Éstos consistían en el modelado de ciertos problemas utilizando redes de Petri, lógicas temporales (implementación en UPPAAL) o álgebras de procesos (implementación en CWS).

2.6.3. Conclusión

La gran ventaja de esta asignatura es que su aplicación se puede realizar en casi cualquier campo, pues algo tan genérico como un sistema (puede ser desde un protocolo utilizado por un router hasta el algoritmo utilizado para aprender un sistema de reglas difusas a partir de una serie de datos) puede ser modelado para la extracción de propiedades del mismo. Es por tanto interesante cursar esta asignatura y conocer las herramientas que estan disponibles para verificar el correcto funcionamiento de un sistema antes de llevarlo a la práctica, y de esta manera evitar errores que pueden resultar más caros (económica o temporalmente) que si se hubiesen detectado con antelación.

Capítulo 3

Aprendizaje de SBRDs: Estado del arte

3.1. Introducción

Para un ser humano, expresar cómo se comporta (o se debe comportar) un sistema es relativamente sencillo. Sin embargo, el experto humano normalmente utiliza el lenguaje natural para expresarlo. Este lenguaje se caracteriza porque la información expresada es imprecisa, como por ejemplo "hace mucho frío", donde el término *mucho frío* es subjetivo y no se puede determinar de manera precisa qué temperatura hace.

Si esta especificación se debe realizar formalmente para que un sistema trabaje con ella de forma automática hay dos opciones: utilizar un lenguaje formal para expresar el comportamiento del sistema o disponer de un mecanismo para representar formalmente la incertidumbre asociada al lenguaje natural. La primera vía hace que el experto humano tenga que utilizar un lenguaje diferente al natural para expresar el conocimiento, lo cual puede reducir su expresividad y hacer que el modelo no se comporte exactamente como se desearía. Es por tanto interesante utilizar la segunda opción.

Junto con las Redes Bayesianas, la lógica difusa es una de las técnicas que permite el tratamiento de la incertidumbre asociada a la información. Esta técnica se apoya en los conjuntos difusos [16], los cuales permiten asociar un grado de verdad a la información. Por ejemplo, se puede decir que hace frio con un grado de verdad de 0.7. En este sentido la lógica difusa ha adquirido gran importancia ya que al permitir trabajar con incertidumbre, hace que los sistemas sean más fáciles de interpretar y de diseñar por estar un nivel más cerca del lenguaje natural.

Siendo capaces de representar esta incertidumbre, una opción para plasmar el comportamiendo de un sistema es mediante reglas del tipo *SI-ENTONCES*. Estos sistemas son conocidos como sistemas basados en reglas difusas.

En este capítulo se desarrollarán conceptos básicos sobre lógica difusa así como

el funcionamiento de ciertos sistemas que basan su funcionamiento en ésta. En primer lugar describiremos los conjuntos difusos en detalle. Tras esto se procederá a describir las reglas difusas y por último se hablará de los sistemas basados en reglas difusas (incluyendo una variante de éstos llamados sistemas basados en reglas difusas lingüísticas). Una vez definida la manera en la que se pueden construir estos sistemas, se verá la forma en la que éstos se pueden aprender de forma automática.

3.2. Conjuntos difusos

En la teoría clásica de conjuntos, dado un elemento $x \in \mathcal{U}$, éste puede pertenecer o no al conjunto $\mathcal{C}$ según la función $f_C(x)$:

$$f_C(x) : X \to \{0, 1\}$$

donde

$$f_C(x) = \begin{cases} 1, & \text{si } x \in C \\ 0, & \text{si } x \notin C \end{cases}$$

Sin embargo, en la teoría de conjuntos difusos [16] se presenta una función de pertenencia no tan estricta, en la que un elemento $x \in \mathcal{U}$ pertenece a un conjunto $\mathcal{C}$ con un grado de verdad $\in [0, 1]$. En la figura 3.1 se puede observar cómo se asigna un cierto grado de verdad $\in [0, 1]$ a un rango de valores $\in [10.0, 50.0]$.

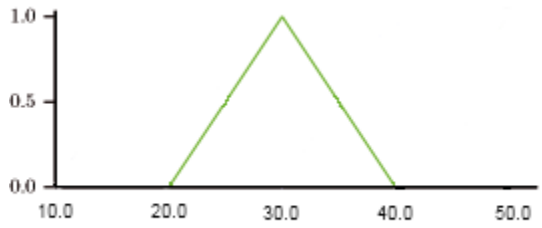

Figura 3.1: Conjunto difuso triangular.

Un conjunto se define mediante una función de pertenencia. Por ejemplo, en la ecuación 3.1 se muestra la función S, y en la ecuación 3.2 la función Z, que son dos funciones muy extendidas. En la imagen 3.2 se puede observar un conjunto difuso definido mediante una función $S(x, 15, 20, 25)$ y otro mediante una función $Z(x, 15, 20, 25)$.

$$S(x; \alpha, \beta, \gamma) = \begin{cases} 0 & \text{si } x \leq \alpha \\ 2 \cdot \left(\frac{x-\alpha}{\gamma-\alpha}\right)^2 & \text{si } \alpha \leq x \leq \beta \\ 1 - 2 \cdot \left(\frac{x-\gamma}{\gamma-\alpha}\right)^2 & \text{si } \beta \leq x \leq \gamma \\ 1 & \text{si } x \geq \gamma \end{cases} \tag{3.1}$$

$$Z(x;\alpha,\beta,\gamma) = 1 - S(x;\alpha,\beta,\gamma) \tag{3.2}$$

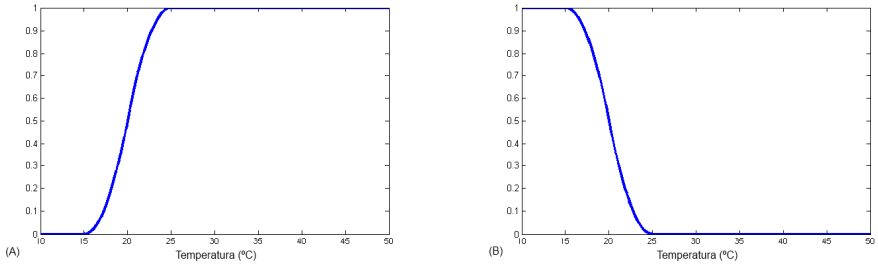

Figura 3.2: Conjunto difuso definido mediante una función S (A) y mediante una función Z (B).

Sin embargo, generalmente se suelen utilizar funciones triangulaes o trapezoidales (como la ilustrada en la figura 3.3), debido a que simplifican determinados cálculos llevados a cabo en un procedimiento denominado proceso de inferencia.

En este sentido, existe una alternativa para expresar la función de pertenencia de una forma más intuitiva. Ésta consiste en definir el valor de los grados de pertenencia para un conjunto finito de puntos, y suponer que la función cambia de modo lineal entre cada par consecutivo de éstos. Así la función de pertenencia $\mu_A(x)$ queda definida de la siguiente manera:

$$\mu_A(x) = \{(\mu_A(x_1)/x_1), (\mu_A(x_2)/x_2), \ldots, (\mu_A(x_n)/x_n)\}\,.$$

Por ejemplo, la función $\mu_A(x) = \{(0.0/15), (1.0/25), (1.0/50)\}$ representa el conjunto difuso ilustrado en la figura 3.3.

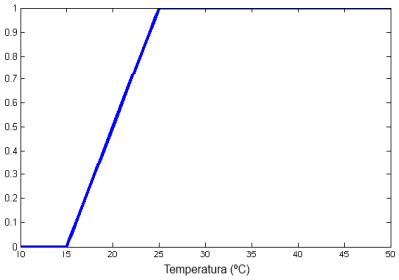

Figura 3.3: Conjunto difuso definido mediante la función $\mu_A(x)$.

Ya que la lógica difusa es una extensión de la lógica clásica, también existen operadores para trabajar sobre conjuntos difusos. Los operadores básicos son la *unión*, la *intersección* y el *complementario*. Su definición es la siguiente:

- *Unión*: Dados dos conjuntos difusos, A y B, su unión $A \cup B$ es otro conjunto difuso tal que:

$$\mu_{A\cup B}(x) = \text{máx}[\mu_A(x), \mu_B(x)]$$

- *Intersección*: Dados dos conjuntos difusos, A y B, su intersección $A \cap B$ es otro conjunto difuso tal que:

$$\mu_{A \cap B}(x) = \text{mín}[\mu_A(x), \mu_B(x)]$$

- *Complementario*: Dado un conjunto difuso, A, su complementario $\overline{A}$ es otro conjunto difuso tal que:

$$\mu_{\overline{A}}(x) = 1 - \mu_A(x)$$

El resultado de cada una de estas operaciones se puede observar en la figura 3.4.

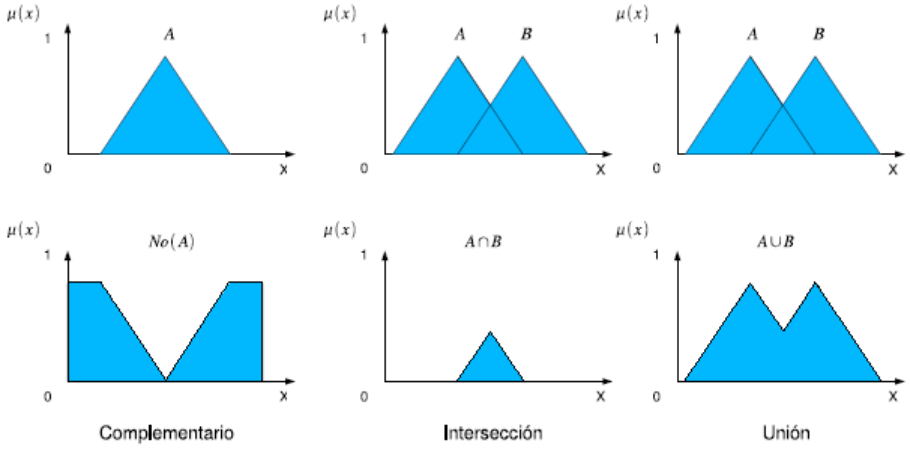

Figura 3.4: Resultado de aplicar las operaciones de unión, intersección y negación (fila inferior) sobre los conjuntos dados en la fila superior.

Existen otras definiciones alternativas para los operadores de *unión* e *intersección*. Las correspondientes a la intersección son denominadas *T-Normas* y las correspondientes a la unión son denominadas *S-Conormas*. Los pares más utilizados de *T-Normas* y *S-Conormas* son los mostrados en el cuadro 3.1.

T-Norma	S-Conorma
mín	máx
producto	máx
producto	$suma - producto$

Cuadro 3.1: Pares de T-normas y S-Conormas comúnmente utilizadas.

3.3. Reglas difusas

Una proposición es una afirmación del tipo (X es F), donde X es una variable en un dominio y F es un conjunto difuso sobre el dominio en el que se ha definido

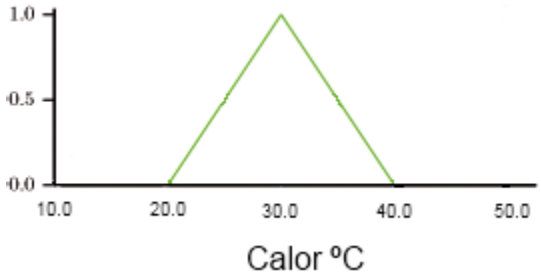

Figura 3.5: Conjunto difuso de la variable *Calor*.

dicha variable. Por ejemplo, (X es F) donde X es la variable *Calor*, y F es el conjunto difuso de la figura 3.5.

Las proposiciones pueden estar unidas por diferentes operadores: $NO(\neg)$, $Y(\wedge)$ y $O(\vee)$. Dadas dos proposiciones X es A e Y es B, y sea $x \in X$ e $y \in Y$, cada operador se define de la siguiente manera:

- NO

$$\mu_{\neg A}(x) = \mu_{\overline{A}}(x) = 1 - \mu_A(x)$$

- Y

$$\mu_{A \wedge B}(x, y) = \mu_{A \cap B}(x, y) = T[\mu_A(x), \mu_B(y)]$$

- O

$$\mu_{A \vee B}(x, y) = \mu_{A \cup B}(x, y) = S[\mu_A(x), \mu_B(y)]$$

donde T y S son una T-Norma y una S-Conorma respectivamente.

Una regla está compuesta de proposiciones unidas por operadores, de la forma:

$$SI\ (\neg)?(X_1 \text{ es } A_1)\ \{\wedge|\vee\} \ldots (\neg)?(X_m \text{ es } A_m)$$
$$ENTONCES\ (\neg)?(X_{m+1} \text{ es } A_{m+1})\ \{\wedge|\vee\} \ldots (\neg)?(X_n \text{ es } A_n)$$

Por tanto, además de los operadores citados anteriormente, es necesario definir la relación de implicación ($\rightarrow$) expreseada anteriormente mediante el término "*ENTONCES*". En la literatura existen principalmente cinco definiciones [8]:

- Larsen: $\mu_{A \rightarrow B}(x, y) = \mu_A(X) \cdot \mu_B(y)$
- Mamdani: $\mu_{A \rightarrow B}(x, y) = \text{mín}\{\mu_A(X), \mu_B(y)\}$
- Lukasiewicz: $\mu_{A \rightarrow B}(x, y) = \text{mín}\{1, 1 - \mu_A(X) + \mu_B(y)\}$
- Zadeh: $\mu_{A \rightarrow B}(x, y) = \text{máx}\{\text{mín}\{\mu_A(X), \mu_B(y)\}, 1 - \mu_A(x)\}$
- Diens-Rescher: $\mu_{A \rightarrow B}(x, y) = \text{máx}\{1 - \mu_A(x), \mu_B(y)\}$

El operador de implicación es lo que se conoce como una relación difusa. Dados dos universos de discurso $\mathcal{U}$ y $\mathcal{V}$, una relación difusa se define como:

$$R : \mathcal{U} \times \mathcal{V} \rightarrow [0, 1]$$

Por ejemplo, si tenemos la definición del conjunto difuso A con la siguiente función de pertenencia

$$\mu_A(x) = \{(0.0/15), (1.0/25), (1.0/50)\}$$

y la definición del conjunto difuso B con la siguiente función de pertenencia $\mu_B(y) = \{(0.0/0), (1.0/5), (0.0/10)\}$, la relación $\mu_{A \rightarrow B}(x, y)$ se expresa, por ejemplo utilizando el operador Mamdani, de la siguiente manera:

$$\mu_{A \rightarrow B}(x, y) = \left| \begin{array}{ccc|c} 0 & 5 & 10 & \\ 0.0 & 0.0 & 0.0 & 15 \\ 0.0 & 1.0 & 0.0 & 25 \\ 0.0 & 1.0 & 0.0 & 50 \end{array} \right. \tag{3.3}$$

Nótese que en caso de tener N proposiciones en el antecedente, la relación sería N-dimensional:

$$R : \mathcal{U}_1 \times \cdots \times \mathcal{U}_N \rightarrow [0, 1]$$

Esta relación esta definida para reglas con una única proposición en el consecuente. Sin embargo, dada una regla con N proposiciones en el consecuente, ésta se puede representar mediante N reglas con un único consecuente, todas ellas con el mismo antecedente pero distinto consecuente $(B_1, \ldots, B_N)$.

3.3.1. Inferencia

Una vez definidas formalmente las reglas difusas, se explicará cómo se puede inferir información a través de ellas. El mecanismo para hacer esto se basa en una extensión del *Modus Ponens* utilizado en la lógica clásica, denominado *Modus Ponens Difuso*.

Dada la regla *SI* (X es A) ENTONCES (Y es B), y la proposición (X es A'), donde A' puede ser distinto de A, puede concluirse que (Y es B'), donde B' también puede ser distinto de B.

El objetivo de la inferencia es, dados $\mu_{A \rightarrow B}(x, y)$ y $\mu_{A'}(x)$, obtener $\mu_{B'}(y)$. Dada la relación $R = \mu_{A \rightarrow B}(x, y)$, B' se obtiene de la siguiente manera:

$$B' = A' \circ R$$

Para resolver esta composición se utiliza la *Regla Composicional de Inferencia*, que se formula a partir del *Principio de Extensión de Zadeh* [18] como:

$$\mu_{B'}(y) = \text{máx}(T[\mu_{A'}(x), \mu_{A \rightarrow B}(x, y)]) \tag{3.4}$$

Por ejemplo, dada la definición del conjunto difuso A'

$$\mu_{A'} = \{(0.0/10), (1.0/15), (0.0/25)\}$$

y los conjuntos difusos A y B utilizados en la ecuación 3.3, habría que calcular $\mu_{B'}(y)$ para todos los puntos en $\mathcal{U} \times \mathcal{V}$. Sin embargo, en este ejemplo en concreto sólo es necesario calcularlo en los pares de puntos {15,20,25} × {0,2.5,7.5,10}:

$$\mu_{A\to B}(x,y) = \left|\begin{array}{cccc|c} 0 & 2.5 & 7.5 & 10 & \\ 0.0 & 0.0 & 0.0 & 0.0 & 15 \\ 0.0 & 0.5 & 0.5 & 0.0 & 20 \\ 0.0 & 0.5 & 0.5 & 0.0 & 25 \end{array}\right.$$

$$\{1.0, 0.5, 0.0\} \circ \left|\begin{array}{cccc|c} 0 & 2.5 & 7.5 & 10 & \\ 0.0 & 0.0 & 0.0 & 0.0 & 15 \\ 0.0 & 0.5 & 0.5 & 0.0 & 20 \\ 0.0 & 0.5 & 0.5 & 0.0 & 25 \end{array}\right. = \{0.0, 0.5, 0.5, 0.0\}$$

Por lo que el resultado de esta inferencia es el conjunto difuso:

$$\mu_{B'} = \{(0.0/0), (0.5/2.5), (0.5/7.5), (0.0/10)\}$$

Obtener todos los pares de puntos para los que es necesario calcular $\mu_{B'}(y)$ en ocasiones no es tan sencillo. Sin embargo, un tipo concreto de inferencia llamada *Inferencia tipo Mamdani* permite obtener estos puntos de forma muy sencilla, lo que hace que sea un tipo de inferencia muy utilizado en la práctica [10, 9]. Esos tipos de inferencia utilizan implicaciones tipo Mamdani junto con el par máx − mín como T-Norma y S-Conorma. La forma de obtener $\mu_{B'}(y)$ es la observada en la figura 3.6. En ella se puede ver como $\mu_{B'}(y) = \text{mín}[\mu_B(y), \text{máx}(\mu_{A\cap A'}(x))]$.

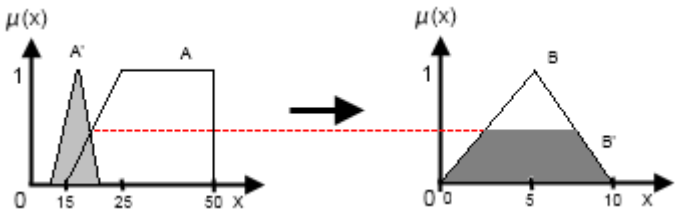

Figura 3.6: Proceso de inferencia tipo Mamdani.

La regla de inferencia descrita en la ecuación 3.4 puede generalizarse para el caso de tener una regla con varias proposiciones en el antecedente unidas por el operador ∧ u ∨, SI (X_1 es A_1) $\wedge/\vee \ldots \wedge/\vee$ (X_m es A_m) ENTONCES (Y es B), de esta forma:

$$\mu_{B'}(y) = \max_{x_1, x_2 \ldots x_n} [\odot(\mu_{A'_1}(x_1), \mu_{A'_2}(x_2), \ldots, \mu_{A'_n}(x_n), \mu_{A\to B}(x_1, x_2, \ldots, x_n, y))]$$

Computacionalmente, calcular $\mu_{B'}(y)$ es muy costoso. Para almacenar la definición del operador de implicación es necesaria una matriz $(N+1)$-dimensional, además de las operaciones necesarias para calcularla. Sin embargo, utilizando la inferencia tipo Mamdani (operador de implicación Mamdani y el par máx − mín como T-Norma y S-Conorma) se consigue simplificar este proceso. En el caso de

reglas con varias proposiciones en el antecedente, en primer lugar se calculan los niveles de corte para cada una de ellas y después se elige uno de entre ellos, dependiendo del operador que una dichas proposiciones. Si están conectados por el operador $\wedge$ se elegirá el menor de todos los niveles de corte. Si el operador utilizado es $\vee$ se elegirá el mayor. Por ejemplo, si tenemos la regla

$$\text{SI } X_1 \text{ es } A_1 \wedge X_2 \text{ es } A_2 \text{ ENTONCES } Y \text{ es } B$$

entonces, el proceso de inferencia da como salida el conjunto difuso mostrado en la figura 3.7.

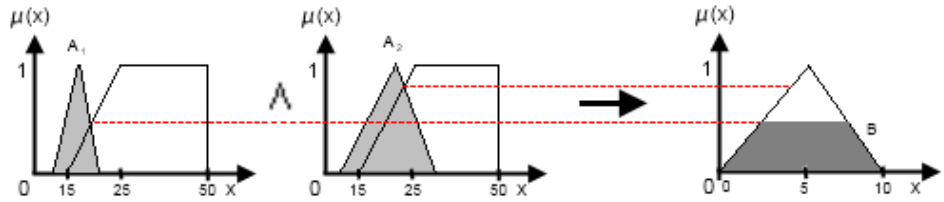

Figura 3.7: Proceso de inferencia tipo Mamdani con varias proposiciones en el antecedente.

Cuando se realiza el proceso de inferencia es posible hacerlo tratando con varias reglas a la vez. Sin embargo, aplicar el operador de implicación en este caso puede resultar un proceso muy complejo, ya que es necesario calcular la relación difusa que representa toda la base de reglas para poder obtener la salida. Una alternativa a esto consiste en aplicar este operador a cada una de las m reglas individualmente, agregando posteriormente cada uno de los m conjuntos difusos obtenidos en salida de este proceso. Por tanto, si se han generado m conjuntos difusos $B'_1, \ldots, B'_m$, el resultado final de la inferencia será:

$$B' = B'_1 \cup B'_2 \cup \ldots \cup B'_m \tag{3.5}$$

3.4. Sistemas basados en reglas difusas

Los sistemas basados en reglas difusas (SBRDs) [17] son usados normalmente para reaccionar ante una entrada numérica, produciendo una salida del mismo tipo. Por ejemplo, para controlar la temperatura de un recinto, se reciben como entradas numéricas la temperatura y la humedad del mismo, y como salida se requiere un valor también numérico que indique la potencia a la que el sistema de aire acondicionado debe funcionar. Estos sistemas se componen de un determinado número de reglas difusas, las cuales utilizan como entrada conjuntos difusos. Por tanto, en un SBRD es necesario realizar una serie de procedimientos para tratar datos numéricos. Éstos son:

1. **Fuzzificación**

   Los datos de entrada suelen ser datos precisos, mientras que el proceso de inferencia recibe como datos de entrada conjuntos difusos. Por ello, es necesario transformar previamente (fuzzificar) los datos precisos en conjuntos difusos.

   Una de las opciones más comunes consiste en generar conjuntos difusos *singleton* a partir de los datos precisos x_n. Dado un $r \in \mathcal{R}$, la función de pertenencia para este tipo de conjuntos viene definida por la siguiente expresión:

$$\mu_S(x) = \begin{cases} 1, & \text{si } x = r \\ 0, & \text{si } x \neq r \end{cases}$$

   Por tanto, un conjunto *singleton* generado a partir de un dato preciso de entrada x_i es aquel cuya función de pertenencia asigna el valor de verdad 1 cuando $x = x_i$ y 0 en caso contrario. Otra opción muy utilizada en la literatura consiste en generar un conjunto definido por una función de pertenencia triangular, de tal manera que para cada dato preciso de entrada x_i:

$$\mu_{A'_i} = \{(0.0/r - \tau), (1.0/r), (0.0/r + \tau)\}, \text{ donde } \tau << |\mathcal{U}|$$

2. **Inferencia**

   Una vez que los valores de entrada x_i han sido fuzzificados, obteniendo los correspondientes conjuntos difusos A'_n, es posible realizar el proceso de inferencia tal y como se explicó en el capítulo 3.3.1. Normalmente la evaluación de cada regla se suele hacer de forma independiente, por lo que para una misma variable de salida Y pueden generarse como resultado de este proceso m conjuntos difusos de salida $B'_1 \dots B'_m$.

3. **Agregación de consecuentes**

   Una vez generados los m conjuntos de salida, es necesario agregarlos (ver ecuación 3.5) para obtener un único conjunto difuso como resultado del proceso de inferencia.

4. **Defuzzificación**

   Al igual que la entrada era un valor preciso, la salida también debe serlo. Por ello es necesario aplicar un proceso inverso a la fuzzificación, llamado defuzzificación, para transformar el conjunto difuso B' a un dato preciso.

   Existen varias alternativas, si bien la más utilizada consiste en calcular el centro de masas o centroide de B', mediante la siguiente expresión:

$$y' = \frac{\int y \cdot \mu_{B'}(y) dy}{\int \mu_{B'}(y) dy}$$

Sin embargo, existe una alternativa en la que los pasos 3 y 4 pueden combinarse en uno solo. Este método se denomina "Modo FITA". En esta alternativa, tras efectuar la inferencia y disponer de m conjuntos difusos como salida de la misma, se procede a defuzzificar cada uno de ellos de forma individual, y el resultado final será la media de todos estos datos precisos. Este proceso está detallado en la siguiente ecuación:

$$y' = \frac{w_1 def(B'_1) + \cdots + w_m def(B'_m)}{w_1 + \cdots + w_m}$$

donde $w_i = 1$.

Se pueden utilizar pesos para ponderar el resultado proporcionado por cada regla en el proceso de inferencia, haciendo que $w_i \in [0, 1]$. De esta manera se puede llegar a conseguir una mayor precisión en el sistema, pero también es cierto que se pierde interpretabilidad por parte de un experto humano.

Respecto a las reglas difusas que componen los SBRDs, existen dos alternativas para su definición. Los conjuntos difusos utilizados en las proposiciones de las que se componen las reglas difusas pueden ser definidos a nivel de sistema o a nivel de regla.

En la primera de estas dos alternativas, existe una definición única para la función de pertenencia asociada a un determinado conjunto difuso. Por ejemplo, dada la proposición (*Calor* es A), la función de pertenencia de A es la función triangular $\mu_A(x) = \{(0.0/20), (1.0/30), (0.0/40)\}$, y cualquier regla que utilice la misma proposición utilizará la misma definición de $\mu_A(x)$.

En la segunda alternativa, a nivel de regla existe una definición única para la función de pertenencia asociada a un determinado conjunto difuso, pero varias definiciones a nivel de sistema. Por ejemplo, dadas las reglas R_1 y R_2, ambas con la proposición (*Calor* es A) en su antecedente, se puede tener la definición $\mu_A(x) = \{(0.0/20), (1.0/30), (0.0/40)\}$ para R_1, y la definición $\mu_A(x) = \{(0.0/25), (1.0/35), (0.0/40)\}$ para R_2.

Mediante el uso de la segunda opción se pueden llegar a conseguir sistemas con una mayor precisión, al dotar de mayor libertad en la definición de las funciones de pertenencia. Sin embargo, esta alternativa resulta mucho menos intuitiva y los sistemas que la utilizan son más difíciles de interpretar por un experto humano.

3.4.1. Sistemas basados en reglas difusas lingüísticas

Con la finalidad de mejorar la legibilidad y la interpretación de sistemas basados en reglas difusas, surgieron los sistemas basados en reglas difusas lingüísticas (SBRDLs) [10]. Cada regla perteneciente a estos sistemas es de la forma (X es A), donde X es una variable lingüística [18], y A es una etiqueta lingüística. Estas etiquetas están asociadas a un conjunto difuso sobre el dominio en el que se ha definido la variable X. Por ejemplo, sea X la variable $Calor$, y A la etiqueta lingüística "High", se está representando mediante esta proposición que "El calor es alto". En la figura 3.8 se muestra 5 posibles etiquetas que podría tomar la variable $Calor$.

Existen ciertas operaciones sobre conjuntos difusos que, utilizadas en sistemas de este tipo, resultan muy intuitivas pues están relacionadas con expresiones del

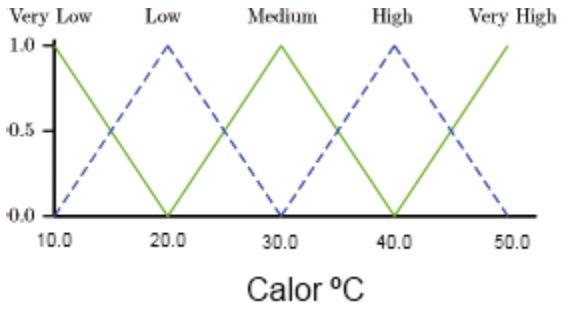

Figura 3.8: Ejemplo de definición de variable lingüística difusa.

lenguaje natural (advervios). Estas operaciones modifican la definición (función de pertenencia) de los conjuntos difusos sobre los que se aplican. Éstas son:

- **Concentración**

  Reduce el grado de pertenencia de los elementos con menor grado. Está relacionado con el advervio "muy" (i.e. "Marta es muy rubia").

$$\mu_{CON(A)} = \mu_A(x)^2, \forall x \in \mathcal{U}$$

- **Dilatación**

  Aumenta el grado de pertenencia de los elementos con menor grado. Está relacionado con el advervio "algo" (i.e. "Marta es algo rubia").

$$\mu_{DIL(A)} = \sqrt{\mu_A(x)}, \forall x \in \mathcal{U}$$

- **Intensificación**

  Reduce el grado de pertenencia de los elementos con grado menor a $\frac{1}{2}$ y aumenta los mayores. Está relacionado con el advervio "bastante"(i.e. "Marta es bastante rubia").

$$\mu_{INT(A)} = \begin{cases} 2^{p-1} \cdot \mu_A(x)^p & \mu_A(x) \leq 0.5 \\ 1 - 2^{p-1} \cdot (1 - \mu_A(x))^p & \mu_A(x) > 0.5 \end{cases}, \forall x \in \mathcal{U}$$

  donde $p > 1$. Normalmente se suele utilizar $p = 2$ (cuanto mayor sea el valor de p, mayor es la intensificación producida).

En la figura 3.9 se muestra de qué manera afectan estos modificadores a un conjunto original (triangular).

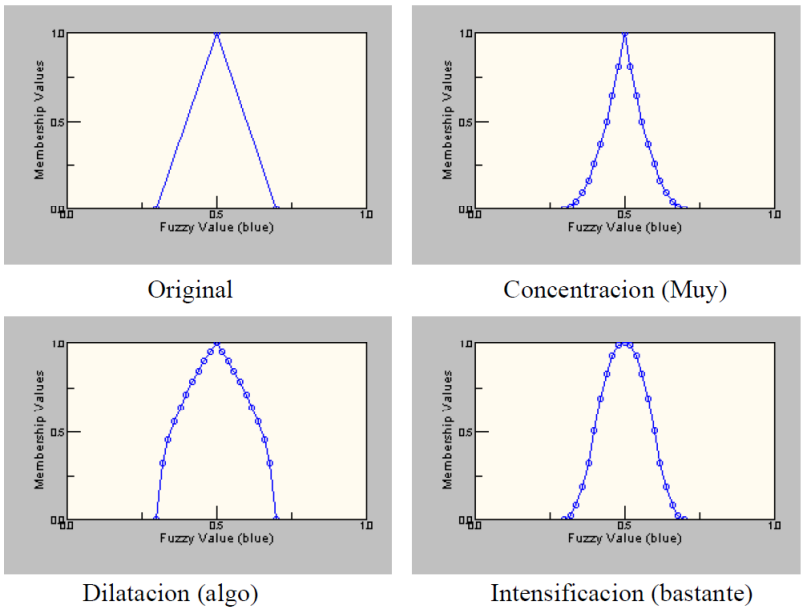

Figura 3.9: Operaciones que modifican la definición de conjuntos difusos.

3.5. Aprendizaje de SBRDL

Se ha visto como los SBRDL presentan ciertas facilidades a la hora de construirlos a partir del lenguaje natural. Sin embargo también es interesante ver de qué manera estos sistemas pueden construirse de forma automática. Los métodos más usados para aprender estos sistemas (llamados *Ad Hoc Data-Driven* [15, 11, 4]) parten de que la definición de las variables, así como de los conjuntos difusos asociados a las mismas han sido dados. Posteriormente, éstos obtienen un conjunto de reglas candidatas (son reglas que potencialmente han podido generar las instancias de la base de datos) tras lo cual se escoge un subconjunto del total. Estas bases de datos están formadas por un número finito de casos, en los cuales se dispone de una combinación de datos de entrada y el valor de salida que el sistema ha generado.

Dependiendo de la forma en la que se obtenga el conjunto de reglas candidatas y el subconjunto final a partir de éste, existen diferentes propuestas. A continuación se describen aquellas más relevantes y que servirán como base para la línea de investigación realizada durante este máster. En primer lugar se hablará sobre el algoritmo de Wang y Mendel [15] y posteriormente se hablará sobre el método COR [2].

3.5.1. Algoritmo de Wang y Mendel

Es una técnica de aprendizaje basada en criterios de envoltura de los datos del conjunto de ejemplos. Tiene la ventaja de que es un proceso bastante rápido aunque en ocasiones la precisión del sistema conseguido puede ser excesivamente baja.

Reglas candidatas

Para generar el conjunto de reglas candidatas, dado un ejemplo $x_1, \ldots, x_n, y$, éste generará una regla de la siguiente forma:

$$\text{SI } (X_1 \text{ es } A'_1) \wedge \ldots \wedge (X_n \text{ es } A'_n) \text{ ENTONCES } (Y \text{ es } B')$$

donde $\mu_{A'_i}(x_i) > \mu_{A_{ij}}(x_i)$, siendo $A'_i \cup A_{ij}$ el conjunto de etiquetas lingüísticas con las que se representa la variable A_i.

Selección de reglas

Durante el proceso de generación de reglas candidatas, es posible que un ejemplo genere una regla con el mismo antecedente que otra ya existente. En este caso, se mantendrá en el conjunto de reglas candidatas una sola de las dos reglas. Ésta (R_s, generada por el ejemplo $e_l = (x_1^l, \ldots, x_n^l, y^l)$) será la que maximice la siguiente expresión (llamada grado de importancia de una regla):

$$id(R_S, e_l) = \mu_{A'_1}(x_1^l) \cdot \ldots \cdot \mu_{A'_n}(x_n^l) \cdot \mu_{B'}(y^l) \tag{3.6}$$

De esta forma, el conjunto de reglas obtenido una vez terminado el proceso de generación de reglas candidatas será el conjunto de reglas del sistema final, en el que no existen dos reglas con el mismo antecedente.

Éste se dice que es un método basado en ejemplos, pues cada ejemplo genera, a lo sumo, una regla. En la figura 3.10 se puede observar como dado el ejemplo e_4, éste generará esta regla candidata:

$$\text{SI } x_1 \text{ es } M \wedge x_2 \text{ es } M \text{ ENTONCES } y \text{ es } B_3$$

De igual manera, el ejemplo e_5 generará otra regla candidata con el mismo antecedente pero diferente consecuente:

$$\text{SI } x_1 \text{ es } M \wedge x_2 \text{ es } M \text{ ENTONCES } y \text{ es } B_2$$

En este caso, se elegirá la regla que maximice la ecuación 3.6 (la regla generada por e_4):

$$\begin{gathered}\mu_M(1.0) \cdot \mu_M(1.2) \cdot \mu_{B_3}(1.6) = 1.0 \cdot 0.8 \cdot 0.6 = 0.48 \\ > \\ \mu_M(1.2) \cdot \mu_M(0.6) \cdot \mu_{B_2}(1.1) = 0.8 \cdot 0.6 \cdot 0.9 = 0.432\end{gathered}$$

3.5.2. Cooperative Rules

Si partimos de un sistema de reglas, la salida ante una entrada proporcionada al mismo no depende de una única regla, sino de la agregación de la salida de varias. Basándose en esta idea, Cooperative Rules (COR) [2] busca una alternativa al aprendizaje de reglas a nivel individual, tal y como lo hace el algoritmo de Wang y Mendel.

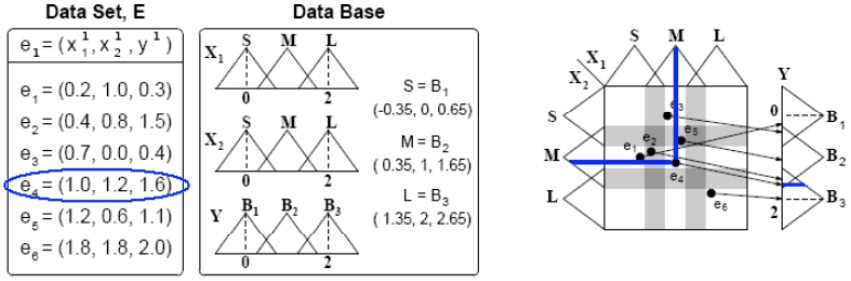

Figura 3.10: Reglas candidatas en el algoritmo de Wang y Mendel.

Reglas candidatas

En primer lugar se clasifican todos los posibles subespacios de entrada que cubren todas las posibles combinaciones de proposiciones en el antecedente para todas las instancias de la base de datos. Cada uno de estos subespacios S_s generará un conjunto de reglas candidatas $CR(S_s)$ con el mismo antecedente pero diferente consecuente. Por ejemplo, dada la base de datos y la definición de las variables difusas de la figura 3.11, existen 9 posibles subespacios de entrada correspondientes a las posibles combinaciones de proposiciones en el antecedente de una regla (en la figura está remarcado el subespacio S_s correspondiente a la combinación de proposiciones en el antecedente M y M, el cual contiene a los ejemplos e_2, e_4 y e_5).

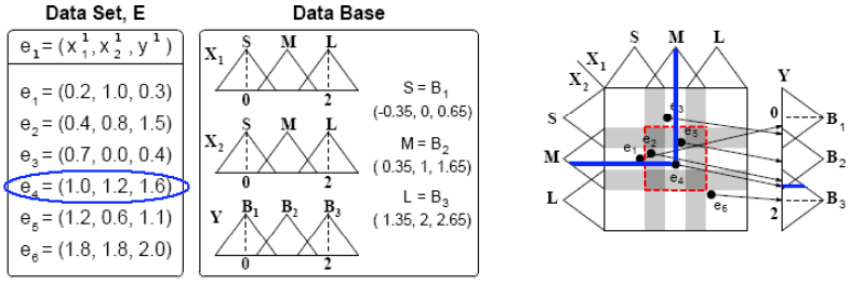

Figura 3.11: Construcción de reglas candidatas en COR.

En la figura 3.12 se ilustran los pasos llevados a cabo para obtener el conjunto de reglas candidatas.

En primer lugar hay que obtener S^+. Éste es un conjunto que contiene todos aquellos subespacios tales que cubren algún ejemplo positivo. Existen dos alternativas para la definición del conjunto de ejemplos positivos dado un subespacio:

1.a Son todos aquellos ejemplos de la base de datos tales que su grado de cobertura en el conjunto S_s (ver ecuación 3.7) es la mayor posible entre el conjunto de grados de cobertura para todos los subespacios, $cd(S_t, e_l), \forall S_t$.

Fase de construcción del espacio de búsqueda

1. *Definición del conjunto de subespacios que cubren el conjunto total de ejemplos* ***E****:* S^+.

   Sea $S_s = (A_1^s, \ldots, A_i^s, \ldots, A_n^s)$, con $A_i^s \in \mathcal{A}_i$,
   $s \in \{1, \ldots, N_s\}$, y $N_s = \prod_{i=i}^{n} |\mathcal{A}_i|$ el número de subespacios. El conjunto de ejemplos positivos para cada subespacio, $(E^+(S_s))$, puede ser definido de dos formas:

   1.a) $(E^+(S_s)) = \{e_l \in \mathbf{E} | \forall i \in \{1, \ldots, n\},$
   $\forall A_i' \in \mathcal{A}_i, \mu_{A_i^s}(x_i^l) \geq \mu_{A_i'}(x_i^l)\}$

   1.b) $(E^+(S_s)) = \{e_l \in \mathbf{E} | \mu_{A_1^s}(x_1^l) \cdot \ldots \cdot \mu_{A_n^s}(x_n^l) \neq 0\}$.

   El conjunto de subespacios de entrada positivos se define de la siguiente forma:

   $$S^+ = S_s | E^+(S_s) \neq \emptyset$$

2. *Generación del conjunto de reglas candidatas en cada subespacio que contiene ejemplos positivos.*

   Cada $S_s \in S^+$ genera un conjunto de reglas candidatas $CR(S_s)$ tal que si

   $$R_{s_k} = [\text{Si } X_1 \text{ es } A_1^s \wedge \ldots \& X_n \text{ es } A_n^s \text{ entonces } Y \text{ es } B_k]$$

   entonces

   $$CR(S_s) = \{R_{s_k} | B_k \in C(S_s)\}$$

   donde $C(S_s)$ es el conjunto de posibles consecuentes para el antecedente generado a partir de S_s, el cual puede ser obtenido también de dos formas:

   2.a) $C(S_s) = \{B_k \in \mathcal{B} | \exists e_l \in E^+(S_s)$
   tal que $\forall B' \in \mathcal{B}, \mu_{B_k}(y^l) \geq \mu_{B'}(y^l)\}$

   2.b) $C(S_s) = \{B_k \in \mathcal{B} | \exists e_l \in E^+(S_s)$
   tal que $\mu_{B_k}(y^l) \neq 0\}$.

   El espacio de búsqueda está definido como todos los conjuntos de reglas RB_o tal que contienen una regla de cada $CR(S_S)$. Por lo tanto,

   $$RB_o =$$

$$\{R_1^o \in CR(S_1), \ldots, R_s^o \in CR(S_s), \ldots, R_{|S^+|}^o \in CR(S_{|S^+|})\}$$

Figura 3.12: Construcción del espacio de búsquerda en COR

$$cd(S_s, e_l) = \mu_{A_1^s}(x_1^l) \cdot \ldots \cdot \mu_{A_n^s}(x_n^l) \tag{3.7}$$

1.b Son todos aquellos ejemplos de la base de datos tales que $cd(S_s, e_l) > 0$. Es decir, S_s son subespacios que al menos cubren un ejemplo e_l.

El siguiente paso consiste en obtener el conjunto de reglas candidatas correspondiente a cada subespacio $S_s \in S^+$. Cada regla obtenida partir de S_s tiene el mismo antecedente (definido por el propio subespacio) pero difiere en el consecuente. El conjunto de posibles consecuentes dado un subespacio S_s puede ser obtenido de dos formas:

2.a Son todos aquellos consecuentes $B_k \in \mathcal{B}$ tales que, dado un ejemplo positivo $e_l \in E^+(S_s)$, no existe otro consecuente B' tal que el grado de verdad $\mu_{B'}(y^l)$ sea mayor o igual que $\mu_{B_k}(y^l)$.

2.b Son todos aquellos consecuentes $B_k \in \mathcal{B}$ tales que, dado un ejemplo positivo $e_l \in E^+(S_s)$, el grado de verdad de $\mu_{B_k}(y^l)$ sea mayor que 0.

Finalmente, el espacio de búsqueda estará formado por todos aquellos conjuntos de reglas RB_o tal que contienen una regla de cada $CR(S_s)$. Normalmente, la propuesta 1.a se combina con la 2.a, y la 1.b con la 2.b. En el primer caso se generan muchas menos reglas, pero la precisión del sistema alcanzado tiende a ser menor.

Por ejemplo, dada la base de datos y la definición de las variables difusas de la figura 3.11 y utilizando la propuesta 1.b con la 2.b, se generarán los subespacios S_1, S_2, S_3 y S_4, cada uno de los cuales tiene como posibles consecuentes $\{B_1, B_2\}$, $\{B_1, B_2, B_3\}$, $\{B_2, B_3\}$ y $\{B_3\}$ respectivamente, tal y como se muestra en la figura 3.13.

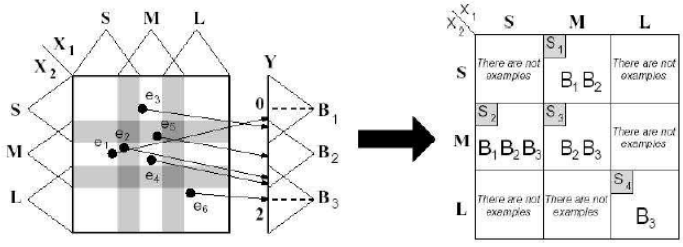

Figura 3.13: Construcción de reglas candidatas en COR utilizando la propuesta 1.b-2.b.

Selección de reglas

En COR la selección de reglas se convierte en un problema de búsqueda. Si $|C(S_S)|$ es el número de posibles consecuentes del subespacio S_s, entonces una posible solución (una determinada selección de reglas) se podría representar mediante un vector de valores discretos $C_o = \{c_1^o, \ldots, c_s^o, \ldots, c_{|S^+|}^o\}$, donde

$c_s^o \in \{0, \ldots, |C(S_s)|\}$. En dicha representación, si $c_s^o > 0$ significa que la regla construida a partir del subespacio S_s con el consecuente c_s^o (consecuente s-ésimo en $|C(S_S)|$) es seleccionado e incluido en la solución final, mientras que si $c_s^o = 0$ significa que la regla construida a partir del subespacio S_s no será incluida en la solución.

Además de la representación de los individuos es necesaria una métrica para poder evaluarlos. Para ello, para cada instancia $e_l = \{x_1^l, \ldots, x_n^l, y^l\}$ perteneciente a la base de datos E se puede medir cualquier tipo de error, como el absoluto (ver ecuación 3.8), el error cuadrático medio (ver ecuación 3.9), etc., entre la salida y'^l generada por el individuo ante la entrada $\{x_1^l, \ldots, x_n^l\}$ e y^l.

$$AE_o(E) = \sum_{l=1}^{N} |y_o'^l - y^l| \tag{3.8}$$

$$MSE_o(E) = \sum_{l=1}^{N} \frac{(y_o'^l - y^l)^2}{N} \tag{3.9}$$

Una vez especificada una codificación para cada individuo (posible solución) y una métrica para evaluar si una solución es mejor o peor que otra, la búsqueda se puede realizar por cualquier algoritmo de optimización combinatoria, como pueden ser Enfriamiento Simulado [2], Algoritmos Genéticos [3], Algoritmos de Estimación de Distribuciones [6] u Optimización basada en Colonias de Hormigas [1].

Capítulo 4

Líneas futuras de investigación

Como se vio en el capítulo 3.5.2, la metodología COR pretende mejorar las soluciones conseguidas mediante otras técnicas como por ejemplo la de Wang y Mendel. Sin embargo, ésta tambien presenta ciertas carencias. La primera se centra en el conjunto de reglas candidatas. Hay que tener en cuenta que todas las reglas incluidas en este conjunto tienen tantas proposiciones en el antecedente como variables de entrada tienen los ejemplos de la base de datos. Esto es una restricción bastante importante, pues si los ejemplos son de la forma $e_l = \{x_1^l, \ldots, x_n^l, y^l\}$, no se permite generar reglas con menos de n proposiciones en el antecedente. Por ello, una posible mejora a esta técnica consiste en relajar la definición de reglas candidatas y permitir la inclusión de reglas con un número i de proposiciones en el antecedente tal que $1 \leq i \leq n$. Sin embargo, aunque esto puede mejorar notablemente la legibilidad y simplicidad (incluso el error) del sistema de reglas alcanzado, también llevaría a generar un conjunto de reglas candidatas mucho más grande y, por consiguiente, sería necesario un mayor esfuerzo por parte del algoritmo de búsqueda para lograr la solución final.

Relacionado con la línea de investigación mencionada anteriormente, otra posible vía sería la de generar sistemas con un menor número de reglas. En este sentido, el problema puede ser enfocado desde dos puntos de vista. En primer lugar, se pueden eliminar reglas a priori (del conjunto de reglas candidatas), antes de que éstas sean tomadas como entrada por el algoritmo de búsqueda. En este sentido se pueden eliminar aquellas reglas que, a priori, parecen poco prometedoras, o aquellas que parecen demasiado específicas dados los ejemplos de la base de datos. Con este enfoque, además de aumentar la interpretabilidad del sistema de reglas (al reducir el número de reglas del sistema final) se puede conseguir un aumento en la rapidez de ejecución, pues el conjunto de reglas que toma el algoritmo de búsqueda es más pequeño. En segundo lugar, se pueden eliminar reglas a posteriori sobre el conjunto de reglas conseguido por el algoritmo de búsqueda. En este sentido se pueden eliminar aquellas que aportan muy poca información al sistema o que incluso hacen disminuir su error.

Para disminuir el esfuerzo computacional necesario para aprender un sistema de reglas, además de la propuesta citada anteriormente, se pueden probar nuevos algoritmos de búsqueda o nuevas configuraciones sobre los ya utilizados en la literatura. Existe un trabajo presentado en 2010 [5] el cual consigue excelentes resultados. Como alternativa a los algoritmos de búsqueda basados en población, éste utiliza algoritmos de búsqueda local (en concreto en este trabajo se utiliza Hill-Climbing e ILS). En este sentido, se puede analizar el uso de otros algoritmos basados en búsqueda local, como puede ser GRASP [7].

Con el objetivo de mejorar la precisión del sistema final, se puede incidir sobre dos aspectos básicos relativos al algoritmo de búsqueda: el individuo de partida y la métrica de evaluación de cada individuo. Respecto al individuo inicial, se pueden buscar definiciones alternativas de tal forma que las reglas más interesantes desde el punto de vista global del sistema estén incluidas en él. Por otra parte, puede ser interesante a la hora de obtener la medición de un individuo utilizar, además del error del sistema, otros parámetros (número de ejemplos que cubre cada regla, grado de especificidad, etc.).

Todos estos enfoques, además de ser aplicables sobre COR, se pueden utilizar sobre otras técnicas de aprendizaje de sistemas de reglas difusas, como wCOR [6] (una variante de COR que incluye el aprendizaje de pesos asociados a las reglas) o sistemas que generan reglas de tipo TSK [13, 12]. Respecto a la primera técnica, hay que tener en cuenta ciertas restricciones, ya que estos algoritmos realizan una búsqueda en el dominio de los reales (debido a los pesos). Por ello, hay que utilizar otras alternativas a la búsqueda local. Los sistemas de reglas tipo TSK tienen la ventaja de que son sistemas muy precisos, y el inconveniente de que éstos pierden interpretabilidad. Aplicando los diferentes enfoques anteriormente citados sería posible mejorar ambas características sobre los sistemas finales.

Capítulo 5

Aportaciones al problema del aprendizaje de SBRDLs

La investigación realizada durante este curso se ha centrado básicamente en la generación de diferentes técnicas para conseguir mejoras sobre el algoritmo COR. Por un lado, se ha definido una nueva metodología para generar el conjunto de reglas candidatas, permitiendo incluir en éste reglas con un número i de proposiciones en el antecedente tal que $1 \leq i \leq n$. Como se comentó, este cambio produce un efecto negativo, ya que el número de reglas candidatas crece en gran medida, por lo que otra línea de investigación que se ha realizado ha consistido en eliminar un determinado número de reglas tras aplicar el algoritmo de búsqueda. Por último, se ha hecho un estudio sobre el impacto de cambiar la solución inicial que toma el algoritmo de búsqueda entre dos alternativas. Respecto al algoritmo de búsqueda, basándonos en los resultados obtenidos en el artículo [5], nos hemos decantado por utilizar una búsqueda local (Hill-Climbing) debido a la relación de ahorro en esfuerzo computacional y calidad de las soluciones obtenidas. La solución inicial de partida para este algoritmo es, al igual que la utilizada en este artículo, aquella que selecciona una regla $R_o^s \in CR(S_s)$ tal que:

$$\exists e_l \in (E^+(S_s)) | id(R_o^s, e_l) > id(R_p^s, e_m), \forall e_m \in (E^+(S_s))$$

donde $R_p^s \in CR(S_s)$, es decir, se selecciona aquella regla de cada conjunto $CR(S_s)$ que maximiza el grado de importancia.

Por ello, los resultados obtenidos en los experimentos serán comparados con los obtenidos utilizando el algoritmo COR-Local Search propuesto en [5].

Estos diferentes algoritmos han sido probados con 6 diferentes bases de datos de ejemplos, tomadas del repositorio de FMLib[1]. De éstas, 4 han sido generadas artificialmente (*f1,f2,f3* y *f4*) y 2 se corresponden con datos tomados del mundo real (*ele1* y *ele2*).

Respecto a los datos artificiales, en los cuatro casos hay 2 variables de entrada y una variable de salida. Todos ellos han sido generados mediante unas funciones matemáticas uniformemente sobre las dos variables de entrada. A continuación,

[1] `http://decsai.ugr.es/~casillas/fmlib/`

para cada problema, se especifica su función, el rango de los datos para cada variable y el número de ejemplos generados.

- *f1*:

$$F_1(x_1, x_2) = x_1^2 + x_2^2$$
$$x_1, x_2 \in [-5, 5] \text{ y } F_1(\cdot) \in [0, 50]$$
$$\text{Número de ejemplos} = 2101$$

- *f2*:

$$F_2(x_1, x_2) = \frac{x_1 - x_1 \cdot x_2}{x_1 - 2 \cdot x_1 \cdot x_2 + x_2}$$
$$x_1, x_2 \in [0, 1] \text{ y } F_2(\cdot) \in [0, 10]$$
$$\text{Número de ejemplos} = 741$$

- *f3*:

$$F_3(x_1, x_2) = e^{x_1} \cdot \sin {x_2}^2 + e^{x_2} \cdot \sin {x_1}^2$$
$$x_1, x_2 \in [-8, 8] \text{ y } F_2(\cdot) \in [0, 5835.70]$$
$$\text{Número de ejemplos} = 1197$$

- *f4*:

$$F_4(x_1, x_2) = x_1^2 + x_2^2 - \cos 18 \cdot x_1 - \cos 18 \cdot x_2$$
$$x_1, x_2 \in [-1, 1] \text{ y } F_2(\cdot) \in [-2, 3.38]$$
$$\text{Número de ejemplos} = 1849$$

Los dos problemas con datos recogidos del mundo real están relacionados con el mundo de la ingeniería. A continuación se describirá cada uno de ellos.

- *ele1*:

  El problema consiste en encontrar un modelo que relacione la longitud total de la línea de baja tensión instalada en una ciudad con el número de habitantes de la misma y la media de las distancias desde el centro de la ciudad hasta los tres clientes más lejanos. El objetivo de este modelo consiste en calcular la longitud total de la línea eléctrica que es necesario mantener.

  Por lo tanto, se dispone de dos variables de entrada y una de salida. Respecto al rango de los datos para las variables de entrada son $x_1 \in [1, 320]$ y $x_2 \in [60, 1673.33]$, y para la variable de salida $y \in [80, 7675]$. El número de ejemplos es de 495.

- *ele2*: En este caso el problema consiste en calcular los costes de mantenimiento mínimos asociados al modelo óptimo para redes electricas en ciudades. El problema dispone de 4 variables de entrada y una de salida. Las variables de entrada son: la suma de las longitudes de todas las calles de la ciudad, el área total de la misma, el área que está ocupada por edificios y el suministro de energia a la ciudad.

  El dominio para las variables de entrada son $x_1 \in [0.5, 11]$, $x_2 \in [0.15, 8.55]$, $x_3 \in [1.64, 142.5]$ y $x_4 \in [1, 165]$, y para la variable de salida se tiene que $y \in [64.47, 8546.03]$. El número de ejemplos es de 1056.

Para todos los algoritmos se han realizado 30 ejecuciones independientes sobre cada base de datos. En cada ejecución se ha elegido el 80 % de los ejemplos como conjunto de entrenamiento y el 20 % restante para test, usando diferentes semillas en cada ejecución pero la misma entre los diferentes algoritmos. Los resultados serán mostrados en tablas con los siguientes datos: error para el conjunto de entrenamiento, error para el conjunto de test, número de reglas, número medio de proposiciones en el antecedente, número de sistemas evaluados y número de instancias de la base de datos procesadas.

A continuación se expondrán cada una de estas líneas así como un análisis sobre los resultados y las conclusiones obtenidas.

5.1. Generación del conjunto de reglas candidatas

Para efectuar esta generación de reglas, se ha optado por aplicar una variante de la secuencia de pasos 1.b y 2.b detallada en el capítulo 3.5.2. Esta variante difiere con la propuesta anteriormente citada en la definición de los subespacios. En este caso, en lugar de quedar definido por n etiquetas lingüísticas, un subespacio $S_{s'}$ queda definido por un conjunto de etiquetas difusas $(A_{i1}^{s'}, \ldots, A_{im}^{s'})$, donde $1 \leq m \leq n, 1 \leq i_1 \leq n, 1 \leq i_m \leq n$ y $i_1 < i_m$. En este caso, ante el cambio sufrido en el número de etiquetas bajo el cual el subespacio $S_{s'}$ queda definido, la ecuación del grado de cobertura para un subespacio se redefine de la siguiente manera:

$$cd(S_{s'}, e_l) = \mu_{A_{i1}^{s'}}(x_{i1}^l) \cdot \ldots \cdot \mu_{A_{im}^{s'}}(x_{im}^l)$$

Dada la nueva definición de subespacio y su grado de cobertura, el resto de pasos se realizan de forma análoga a lo descrito en el capítulo 3.5.2. A esta versión de COR se la ha denominado Extended COR (ECOR).

Los resultados obtenidos se muestran en el cuadro 5.1.

Como se puede observar, en la mayoria de los casos se mejora el error para el conjunto de test (para todas las bases de datos excepto para *f2*).

Respecto al número de reglas hay que hacer una distinción entre *ele2* y el resto de bases de datos. Para *ele2* el número de variables de entrada es 4, por lo que parece normal que el número de reglas generadas crezca en mayor medida que para aquellos casos en los que el número de variables de entrada es 2. Si sobre un conjunto con un mayor número de reglas se procede a la expansión del mismo según la variante expuesta en el método ECOR, este aumento en % es mucho mayor. Por ello, si bien para *f1, f2, f3 ,f4* y *ele1* la diferencia de reglas entre ambos algoritmos no es muy grande (e incluso en ciertas ocasiones ECOR consigue un menor número de reglas, como por ejemplo para *f3*), para *ele2* la diferencia es mucho más notable, generando más reglas en el caso del algoritmo ECOR.

Como cabía esperar, al aumentar el número de reglas candidatas el algoritmo de búsqueda requiere un mayor esfuerzo computacional para encontrar la solución final por lo que el número de sistemas evaluados y el número de instancias procesadas es mucho mayor en las ejecuciones de este algoritmo.

5.2. Eliminación de reglas

Se ha podido comprobar que al generar un conjunto de reglas candidatas mayor, se tiende a incluir un mayor número de reglas en el sistema final. Este problema cobra más importancia cuanto mayor es el número de variables de entrada. Sin embargo, es posible que muchas de estas reglas sean irrelevantes o aporten muy poca información al sistema.

Por ello, a través de este enfoque se ha pretendido comprobar el efecto de aplicar un postproceso sobre el sistema obtenido tras aplicar el algoritmo de búsqueda consistente en la eliminación de determinadas reglas. Las reglas a eliminar deben ser aquellas que hagan que el sistema empeore lo mínimo posible (o en su caso que mejore lo máximo posible). Respecto al criterio de parada para dejar de eliminar reglas, sea ϵ_0 el error del individuo R_0 antes de aplicar este postproceso, $u \in [0, 1]$ un umbral de eliminación, y ϵ_i el error del individuo R_i obtenido tras eliminar i reglas, si $\epsilon_i \geq \epsilon_0 \cdot (1+u)$ entonces el postproceso de eliminación de reglas finalizará, debiendo continuar por la iteración $i + 1$ en caso contrario. De esta forma, se eliminarán reglas mientras el error del sistema no empeore tanto o más de un $(u \cdot 100)\,\%$ del sistema de partida.

Los resultados obtenidos se muestran en los cuadros 5.2 y 5.3.

Este estudio arroja resultados muy interesantes. En primer lugar, comparando los resultados conseguidos por el algoritmo COR-Local Search sin eliminación de reglas y el ExtendedCOR-LocalSearch eliminando reglas con $u = 0.10$, para algunos casos (para *f1*, *f4* y *ele2*) se puede observar como éste último mejora el error conseguido por el primero incluso con un número bastante menor de reglas. Esto quiere decir que las reglas más generales proporcionan más informacion que las más específicas y permiten construir individuos con un menor error. Respecto a *f2*, tal y como pasaba en el experimento anterior, no se consigue mejorar los resultados del algorimto COR-Local Search. Esto puede ser debido a que los datos se comportan en todos los casos muy dependientemente del valor de ambas variables, por lo que el uso de reglas específicas se hace necesario. Para *f3* se puede observar como el número medio de proposiciones en el antecedente es *NaN*. Esto es debido a que el sistema alcanzado no contiene ninguna regla. Esto sucede así porque cuando se intenta procesar un ejemplo, el cual no dispara ninguna regla, se devuelve un valor como salida igual a la media de todos los valores de salida de los ejemplos del conjunto de entrenamiento. En este caso, si se tienen datos distribuidos de una manera muy uniforme tal que esta "predicción" es relativamente buena, cuando se proceda a la eliminación de las reglas el sistema irá mejorando (en lugar de aumentar el error) hasta el punto de que todas las reglas serán eliminadas, al no conseguir un empeoramiento del sistema de un $(1 + u)\,\%$. Por último, para *ele1* los resultados conseguidos, comparando de igual a igual (COR-LS con ExtendedCOR-LS y eliminando reglas con el mismo parámetro u), son siempre mejores en la versión aquí propuesta que en el algoritmo COR-Local Search.

Además es un dato interesante que al eliminar reglas en el algoritmo Extended-COR-Local Search se tiendan a eliminar reglas más específicas, pues en todos los casos tras aplicar este postproceso el número medio de proposiciones en el antecedente disminuye (mayor es esta disminución cuanto mayor es el valor del parámetro u). Esto quiere decir que las reglas más específicas son las que proporcionan menos información (o información menos relevante) que las más generales.

Por último, como cabía esperar, el número de sistemas evaluados y de instancias procesadas también crece al aplicar este proceso, aunque no en gran medida. Esto es normal pues es necesario evaluar cada individuo eliminando cada vez una de sus reglas, lo cual incrementa estos valores, pero no en tanta medida como la búsqueda local pues en este caso lo único que se realiza es un proceso de eliminación de reglas (no se prueban todas las posibles alteraciones en la representación del individuo sino que cada vez solo se alterará una componente c_s^o del vector $C_o = \{c_1^o, \ldots, c_s^o, \ldots, c_{|S^{++}|}^o\}$ por 0).

5.3. Individuo de partida para el algoritmo de búsqueda

Por último, se ha querido variar la definición del individuo de partida para el algoritmo de búsqueda. Dada la definición de subespacios en COR, elegir como solución inicial el sistema que se compone de las reglas $R_o^s \in CR(S_s)$ que maximizan $cd(R_o^s, S_s), \forall S_s \in S^+$ puede ser una buena opción pues se consigue que el sistema de partida cubra todos los ejemplos de la base de datos. Sin embargo, dada la nueva definición de subespacios, esta solución inicial puede no ser la mejor opción ya que al tener reglas menos específicas (que cubren un mayor número de ejemplos) el sistema inicial partirá de un conjunto de reglas las cuales harán que uno o varios ejemplos queden cubiertos por más de una regla. Es por tanto interesante comprobar el comportamiento del algoritmo ante un individuo de partida con un menor número de reglas. En esta línea se ha elegido como solución inicial el conjunto de reglas vacío, descargando el peso del proceso de selección de reglas en su totalidad sobre el algoritmo de búsqueda local.

Los resultados obtenidos se muestran en los cuadros 5.4 y 5.5.

Como se puede observar, en la mayoria de los casos el error conseguido es peor, salvo para los conjuntos de test *f2* y *f3* utilizando 5 etiquetas. Esto se debe a que alcanza un óptimo local demasiado pronto. Como consecuencia de ello, el número medio de reglas es mucho más bajo y el número de sistemas evaluados e instancias procesadas también disminuye considerablemente.

Otro dato relevante es que el número medio de proposiciones en el antecedente aumenta para la mayoria de casos. Esto es debido a que, de forma individual, las reglas más especificas consiguen disminuir el error en mayor medida que las reglas más generales, produciéndose un sobreajuste del sistema.

Como conclusión final, en este capítulo se ha podido comprobar como la expansión del conjunto de reglas candidatas ha producido una importante mejora en la precisión del sistema alcanzado. Sin embargo, el número de reglas incluidas en éste es elevado. También, como se ha podido observar, un gran número de éstas aportan muy poca información e incluso pueden hacer que la precisión del sistema empeore. Como continuación al trabajo desarrollado durante este curso se pretende investigar una línea con un claro objetivo marcado: intentar descartar estas reglas antes de aplicar la búsqueda local, permitiendo acelerar el proceso de aprendizaje y produciendo sistemas con una mayor precisión. Esta línea se podría desarrollar mediante un método constructivo (como lo es GRASP [7]).

Resultados para la base de datos *f1*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-Local Search	2,614	2,594	25,000	2,000	309,167	70641,300
ExtendedCOR-Local Search	**1,846**	1,849	29,833	1,665	1555,100	553682,200

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-Local Search	1,714	1,771	48,967	2,000	832,467	95337,883
ExtendedCOR-Local Search	1,077	**1,105**	53,800	1,775	3234,067	780518,000

Resultados para la base de datos *f2*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-Local Search	0,603	**0,604**	23,000	2,000	159,000	11918,233
ExtendedCOR-Local Search	1,117	1,124	24,033	1,700	1215,867	153344,467

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-Local Search	0,380	**0,392**	46,867	2,000	461,367	17796,233
ExtendedCOR-Local Search	0,808	0,807	37,200	1,793	4092,933	363967,733

Resultados para la base de datos *f3*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-Local Search	623,012	625,089	17,133	2,000	429,200	44550,400
ExtendedCOR-Local Search	608,672	**614,252**	10,700	1,303	1982,133	393635,300

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-Local Search	484,915	511,837	41,833	2,000	627,167	33441,967
ExtendedCOR-Local Search	465,178	**498,550**	27,100	1,674	3685,833	520573,767

Resultados para la base de datos *f4*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-Local Search	1,019	1,040	20,967	2,000	789,600	155325,233
ExtendedCOR-Local Search	0,992	**1,006**	30,300	1,682	2138,400	656817,100

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-Local Search	1,051	1,103	48,733	2,000	4223,967	440431,733
ExtendedCOR-Local Search	0,917	**0,953**	46,767	1,768	15092,533	3107216,600

Resultados para la base de datos *ele1*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-Local Search	622,735	712,780	18,033	2,000	305,667	21382,767
ExtendedCOR-Local Search	609,895	**689,214**	24,733	1,676	1244,200	126763,233

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-Local Search	567,673	694,345	28,100	2,000	624,300	29393,600
ExtendedCOR-Local Search	564,187	**677,918**	38,200	1,726	2081,800	169427,433

Resultados para la base de datos *ele2*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-Local Search	376,041	391,472	189,600	4,000	16485,167	987854,100
ExtendedCOR-Local Search	343,303	**353,470**	515,100	3,174	323668,234	33930073,867

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-Local Search	242,831	277,375	440,467	4,000	22223,200	833605,767
ExtendedCOR-Local Search	223,465	**236,613**	1109,734	3,232	499315,834	36687634,600

Cuadro 5.1: Comparación de los resultados para COR y ECOR.

Resultados para la base de datos *f1*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	2,614	2,594	25,000	2,000	309,167	70641,300
COR-LS $u = 0.01$	2,667	2,729	24,567	2,000	336,333	77266,233
COR-LS $u = 0.10$	3,339	3,441	20,100	2,000	362,433	82587,133
ExtendedCOR-LS	1,846	**1,849**	29,833	1,665	1555,100	553682,200
ExtendedCOR-LS $u = 0.01$	1,851	1,857	29,167	1,657	1597,833	568266,833
ExtendedCOR-LS $u = 0.10$	1,932	1,949	18,433	1,455	1723,333	616479,533

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	1,714	1,771	48,967	2,000	832,467	95337,883
COR-LS $u = 0.01$	1,746	1,883	48,000	2,000	887,633	102531,533
COR-LS $u = 0.10$	2,079	2,113	42,467	2,000	921,967	106502,867
ExtendedCOR-LS	1,077	**1,105**	53,800	1,775	3234,067	780518,000
ExtendedCOR-LS $u = 0.01$	1,090	1,123	51,667	1,765	3321,667	798173,033
ExtendedCOR-LS $u = 0.10$	1,164	1,190	44,500	1,727	3408,867	818660,733

Resultados para la base de datos *f2*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	0,603	**0,604**	23,000	2,000	159,000	11918,233
COR-LS $u = 0.01$	0,604	0,653	22,567	2,000	185,433	14172,300
COR-LS $u = 0.10$	0,649	0,724	18,833	2,000	207,167	15853,900
ExtendedCOR-LS	1,117	1,124	24,033	1,700	1215,867	153344,467
ExtendedCOR-LS $u = 0.01$	1,118	1,118	20,600	1,651	1286,233	159759,267
ExtendedCOR-LS $u = 0.10$	1,181	1,182	14,600	1,509	1358,533	166474,400

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	0,380	**0,392**	46,867	2,000	461,367	17796,233
COR-LS $u = 0.01$	0,385	0,457	43,667	2,000	527,867	20585,700
COR-LS $u = 0.10$	0,431	0,537	35,200	2,000	591,233	22901,500
ExtendedCOR-LS	0,808	0,807	37,200	1,793	4092,933	363967,733
ExtendedCOR-LS $u = 0.01$	0,811	0,805	31,967	1,758	4216,367	369109,433
ExtendedCOR-LS $u = 0.10$	0,848	0,850	26,267	1,707	4282,100	371853,433

Resultados para la base de datos *f3*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	623,012	625,089	17,133	2,000	429,200	44550,400
COR-LS $u = 0.01$	566,545	562,130	0,133	NaN	557,400	53550,400
COR-LS $u = 0.10$	569,655	565,930	0,000	NaN	557,400	53550,400
ExtendedCOR-LS	608,672	614,252	10,700	1,303	1982,133	393635,300
ExtendedCOR-LS $u = 0.01$	567,902	**559,073**	0,400	NaN	2186,233	407274,600
ExtendedCOR-LS $u = 0.10$	569,655	565,930	0,000	NaN	2192,233	407387,133

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	484,915	511,837	41,833	2,000	627,167	33441,967
COR-LS $u = 0.01$	559,814	558,378	2,267	NaN	936,867	46223,500
COR-LS $u = 0.01$	564,284	562,087	0,533	NaN	949,167	46607,667
ExtendedCOR-LS	465,178	**498,550**	27,100	1,674	3685,833	520573,767
ExtendedCOR-LS $u = 0.01$	520,544	535,922	6,567	NaN	4092,267	541600,800
ExtendedCOR-LS $u = 0.01$	563,463	557,102	0,400	NaN	4193,967	545764,767

Cuadro 5.2: Comparación de los resultados eliminando reglas para COR y ECOR (a).

Resultados para la base de datos *f4*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	1,019	1,040	20,967	2,000	789,600	155325,233
COR-LS $u = 0.01$	1,029	1,048	12,300	2,000	861,800	166255,933
COR-LS $u = 0.10$	1,109	1,118	2,200	NaN	922,233	171430,167
ExtendedCOR-LS	0,992	**1,006**	30,300	1,682	2138,400	656817,100
ExtendedCOR-LS $u = 0.01$	1,000	1,008	18,567	1,488	2307,467	710565,600
ExtendedCOR-LS $u = 0.10$	1,076	1,080	7,867	1,328	2463,567	745776,067

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	1,051	1,103	48,733	2,000	4223,967	440431,733
COR-LS $u = 0.01$	1,059	1,119	42,400	2,000	4309,933	449452,667
COR-LS $u = 0.10$	1,133	1,186	19,767	2,000	4455,333	460399,100
ExtendedCOR-LS	0,917	**0,953**	46,767	1,768	15092,533	3107216,600
ExtendedCOR-LS $u = 0.01$	0,922	0,962	34,733	1,694	15285,967	3137065,900
ExtendedCOR-LS $u = 0.10$	0,991	1,016	19,633	1,525	15482,367	3166173,367

Resultados para la base de datos *ele1*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	622,735	712,780	18,033	2,000	305,667	21382,767
COR-LS $u = 0.01$	626,528	709,766	12,333	2,000	348,033	23601,333
COR-LS $u = 0.10$	690,472	780,361	6,533	2,000	381,333	25434,500
ExtendedCOR-LS	609,895	**689,214**	24,733	1,676	1244,200	126763,233
ExtendedCOR-LS $u = 0.01$	613,013	692,825	15,500	1,523	1358,900	135518,500
ExtendedCOR-LS $u = 0.10$	671,646	739,201	7,167	1,344	1454,600	143463,200

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	567,673	694,345	28,100	2,000	624,300	29393,600
COR-LS $u = 0.01$	569,835	712,502	20,700	2,000	687,967	31449,967
COR-LS $u = 0.10$	620,022	734,674	11,500	2,000	737,467	33131,567
ExtendedCOR-LS	564,187	**677,918**	38,200	1,726	2081,800	169427,433
ExtendedCOR-LS $u = 0.01$	566,947	679,126	26,033	1,627	2237,833	178110,767
ExtendedCOR-LS $u = 0.10$	621,493	734,150	11,367	1,413	2400,733	188289,267

Resultados para la base de datos *ele2*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	376,041	391,472	189,600	4,000	16485,167	987854,100
COR-LS $u = 0.01$	377,154	418,529	113,200	4,000	18379,300	1026707,533
COR-LS $u = 0.10$	395,489	446,747	43,867	4,000	20669,333	1065703,500
ExtendedCOR-LS	343,303	**353,470**	515,100	3,174	323668,234	33930073,867
ExtendedCOR-LS $u = 0.01$	343,963	354,886	265,400	2,903	353776,200	35326454,334
ExtendedCOR-LS $u = 0.10$	360,622	377,136	80,167	2,313	387331,400	36509583,534

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	242,831	277,375	440,467	4,000	22223,200	833605,767
COR-LS $u = 0.01$	243,064	297,085	307,067	4,000	25206,767	860790,600
COR-LS $u = 0.10$	251,719	319,171	133,067	4,000	29424,733	900055,233
ExtendedCOR-LS	223,465	**236,613**	1109,734	3,232	499315,834	36687634,600
ExtendedCOR-LS $u = 0.01$	223,946	237,905	548,500	2,929	563349,800	38711518,100
ExtendedCOR-LS $u = 0.10$	232,443	251,026	195,067	2,464	624633,200	40469200,367

Cuadro 5.3: Comparación de los resultados eliminando reglas para COR y ECOR (b).

Resultados para la base de datos *f1*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	2,614	2,594	25,000	2,000	309,167	70641,300
COR-LS Empty	6,578	6,627	5,000	2,000	269,000	57254,833
ExtendedCOR-LS	1,846	**1,849**	29,833	1,665	1555,100	553682,200
ExtendedCOR-LS Empty	6,346	6,432	5,933	1,967	537,800	187151,133

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	1,714	1,771	48,967	2,000	832,467	95337,883
COR-LS Empty	2,178	2,221	44,400	2,000	2617,600	296751,267
ExtendedCOR-LS	1,077	**1,105**	53,800	1,775	3234,067	780518,000
ExtendedCOR-LS Empty	2,657	2,705	40,600	1,948	4282,833	1086368,133

Resultados para la base de datos *f2*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	0,603	**0,604**	23,000	2,000	159,000	11918,233
COR-LS Empty	0,858	0,883	21,233	2,000	780,533	61238,833
ExtendedCOR-LS	1,117	1,124	24,033	1,700	1215,867	153344,467
ExtendedCOR-LS Empty	0,864	0,887	21,167	1,994	1546,200	214393,800

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	0,380	**0,392**	46,867	2,000	461,367	17796,233
COR-LS Empty	0,581	0,588	42,533	2,000	1992,933	77490,500
ExtendedCOR-LS	0,808	0,807	37,200	1,793	4092,933	363967,733
ExtendedCOR-LS Empty	0,609	0,619	35,200	1,908	3689,200	352076,033

Resultados para la base de datos *f3*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	623,012	625,089	17,133	2,000	429,200	44550,400
COR-LS Empty	546,361	**543,192**	1,033	2,000	120,900	11888,700
ExtendedCOR-LS	608,672	614,252	10,700	1,303	1982,133	393635,300
ExtendedCOR-LS Empty	546,361	**543,192**	1,033	2,000	198,767	34165,967

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	484,915	511,837	41,833	2,000	627,167	33441,967
COR-LS Empty	533,979	539,941	6,900	2,000	448,567	24257,467
ExtendedCOR-LS	465,178	**498,550**	27,100	1,674	3685,833	520573,767
ExtendedCOR-LS Empty	534,067	535,174	5,400	1,972	780,867	97249,500

Cuadro 5.4: Comparación de los resultados para COR y ECOR con el conjunto de reglas inicial vacío (a).

Resultados para la base de datos *f4*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	1,019	1,040	20,967	2,000	789,600	155325,233
COR-LS Empty	1,026	1,030	10,933	2,000	555,333	109355,933
ExtendedCOR-LS	0,992	**1,006**	30,300	1,682	2138,400	656817,100
ExtendedCOR-LS Empty	1,030	1,040	9,400	1,825	920,833	289420,500

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	1,051	1,103	48,733	2,000	4223,967	440431,733
COR-LS Empty	1,012	1,047	39,767	2,000	3021,067	318419,067
ExtendedCOR-LS	0,917	**0,953**	46,767	1,768	15092,533	3107216,600
ExtendedCOR-LS Empty	0,971	1,007	35,833	1,839	6799,033	1440781,100

Resultados para la base de datos *ele1*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	622,735	712,780	18,033	2,000	305,667	21382,767
COR-LS Empty	732,927	819,401	12,933	2,000	455,967	27619,300
ExtendedCOR-LS	609,895	**689,214**	24,733	1,676	1244,200	126763,233
ExtendedCOR-LS Empty	732,177	801,101	13,233	1,861	947,533	91981,300

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	567,673	694,345	28,100	2,000	624,300	29393,600
COR-LS Empty	665,872	765,943	18,833	2,000	753,067	33753,333
ExtendedCOR-LS	564,187	**677,918**	38,200	1,726	2081,800	169427,433
ExtendedCOR-LS Empty	655,692	776,287	23,367	1,871	1982,933	159695,967

Resultados para la base de datos *ele2*

5 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	376,041	391,472	189,600	4,000	16485,167	987854,100
COR-LS Empty	418,342	440,605	76,567	4,000	9458,500	303817,833
ExtendedCOR-LS	343,303	**353,470**	515,100	3,174	323668,234	33930073,867
ExtendedCOR-LS Empty	655,692	776,287	23,367	1,871	1982,933	159695,967

7 etiquetas

Alg. búsqueda	Err. entrenam.	Err. Test	#Reglas	#Prop. antecedente	#Sist. evaluados	#Inst. procesadas
COR-LS	242,831	277,375	440,467	4,000	22223,200	833605,767
COR-LS Empty	300,904	357,294	146,033	4,000	14256,600	269414,333
ExtendedCOR-LS	223,465	**236,613**	1109,734	3,232	499315,834	36687634,600
ExtendedCOR-LS Empty	250,081	271,120	227,633	3,399	142381,267	7607257,733

Cuadro 5.5: Comparación de los resultados para COR y ECOR con el conjunto de reglas inicial vacío (b).

Capítulo 6

Conclusión

El trabajo realizado durante este curso en el máster en Tecnologías Informáticas Avanzadas de la Universidad de Castilla-La Mancha se puede clasificar en dos partes bien diferenciadas (asignaturas y trabajo de investigación), las cuales han sido expuestas en esta memoria.

En primer lugar se han detallado los contenidos de las asignaturas cursadas, así como el trabajo desarrollado para superarlas y la relación e influencia que han tenido con la línea de investigación llevada a cabo.

En segundo lugar, se ha realizado una introducción a la investigación llevada a cabo a través del capítulo 3. En éste se han presentado las nociones básicas necesarias para entender el problema a tratar (qué son las reglas difusas, de qué elementos se componen, cómo se puede sacar partido a las mismas, ...). También se han mencionado los sistemas de reglas difusas más utilizados, y cómo los SBRDL ayudan a la definición y generación de estos sistemas por parte de expertos humanos. Posteriormente, se han hablado de ciertos métodos para la generación de estos sistemas de manera automática a partir de un conjunto de ejemplos. Se ha presentado la problemática asociada a ello y las principales formas de resolverlo, presentando algunos de los algoritmos más importantes diseñados para este fin. Uno de ellos (COR) se ha explicado a un mayor nivel de detalle, pues es el algoritmo básico sobre el que se ha realizado la línea de investigación a lo largo de este curso de máster.

La investigación llevada a cabo durante este curso se ha centrado en mejorar la expresividad de las reglas generadas por la metodología COR. Además, gracias a ello, las metaheurísticas que utiliza COR para llevar a cabo el proceso de búsqueda son capaces de encontrar mejores soluciones antes de estancarse en óptimos locales. A su vez se ha tratado de reducir el problema, amplificado en esta variante de COR, de la inclusión de un elevado número de reglas en el sistema final. Para ello, se han investigado dos alternativas: aplicar un postproceso que se encargue de eliminar aquellas reglas menos relevantes y comenzar la búsqueda local a partir de un individuo que representa el conjunto de reglas vacío.

Con los resultados obtenidos se han abierto otras líneas de investigación que se pretenden abordar durante la realización del doctorado. Como se ha podido

observar, partir del conjunto vacío de reglas no es una buena opción, pero da pie a pensar que partiendo de un conjunto con pocas reglas puede dar buenos resultados. Una alternativa consiste en comenzar la búsqueda con el menor número de reglas tal que todos los ejemplos estén cubiertos. Otra alternativa consiste en buscar un equilibrio entre incluir reglas que cubren un gran número de ejemplos y reglas que provocan una disminución del error del sistema muy alto. En cualquier caso, la generación de este individuo incial se podría realizar mediante un método constructivo (como lo es GRASP).

También cabe destacar que todas estas diferentes vías estudiadas para mejorar el comportamiento de COR, son aplicables a otras técnicas de aprendizaje. Son de especial interes aquellos que generan reglas de tipo TSK, puesto que éstos logran una mayor precisión en comparación con COR. Por ello se pretende estudiar el efecto de estos enfoques sobre los mismos (en concreto sobre aquellos que generan reglas TSK de orden 0, pues son reglas más sencillas con una interpretación más sencilla e intuitiva).

Formación académica

09/2005–09/2010 Ingeniero en Informática. Escuela Suprerior de Ingeniería Informática. Universidad de Castilla-La Mancha.

- Nota media: 8,5 (sobre 10)
- Itinerario: Sistemas Informáticos
- Itinerario adicional: Arquitectura, redes y sistemas de comunicaciones.
  Cursado mediante asignaturas adicionales en la modalidad de libre configuración.

10/2010–? Máster Universitario en Tecnologias Informaticas Avanzadas. Escuela Suprerior de Ingeniería Informática. Universidad de Castilla-La Mancha.

- Máster Oficial con referencia MO2006-00197.
- Incluido en el posgrado con Programa de Doctorado en Tecnologías Informáticas Avanzadas, con Mención de Calidad del Ministerio (MCD2006-00423).

Puestos Ocupados

01/2010–06/2010 Beca de colaboración en el Departamento de Sistemas Informáticos (UCLM). Financiada por el Ministerio de Educación.
Supervisor: Prof. José L. Sánchez.

Resumen: Utilización de CUDA para la paralelización de código en herramientas de procesado de imágenes para la detección temprana de alzheimer y otras demencias, TSI-020110-2009-362, financiado por el Ministerio de Industria, Turismo y Comercio.

10/2010–*04/2011* Beca de Iniciación a la Investigación. Financiada por el Vicerrectorado de Investigación de la UCLM.
Supervisor: Prof. José A. Gámez

Línea de trabajo: Sistemas Inteligentes; Aprendizaje automático; Sistemas basados en reglas difusas.

Premios recibidos

Ganador del segundo premio "Mejor Utilidad" en el III Concurso Universitario de Software Libre de Castilla La-Mancha (cuso 2009/2010), con el desarrollo del proyecto TURMS.

Autores Hugo Caballero, Javier Cózar y José M. Serrano

Resumen: TURMS es una aplicación para empresas, que permite a sus clientes(mediante conexión wifi y un navegador web) buscar productos, información sobre los mismos, cómo llegar a ellos, etc. Se dispone de una aplicación de escritorio para permitir modelar el negocio y especificar los productos.

Actividad Investigadora

07/2010	Participación en la *ESTRO International Oncology Forum: Clinical Perspectives in Radiation Oncology (2010)*, presentando una paralelización de un software clínico dedicado a la gestión de tratamientos radiológicos en oncología utilizando GPUs mediante la arquitectura CUDA (NVIDIA).
10/2010–?	Colaborador en el proyecto de investigación *PROGRAMO: Diseño de nuevos algoritmos en modelos gráficos probabilísticos. Implementación en Elvira.* (TIN2007-67418-C03-01). Financiado por el Plan Nacional. De Octubre de 2007 a Marzo de 2011. Investigador principal: Dr. José A. Gámez.
10/2010–?	Colaborador en el proyecto de investigación *Desarrollo de nuevas técnicas metaheurísticas y apicaciones a problemas de minería de datos.* (PCI08-0048-8577). Financiado por el Plan Regional (JCCM). De Enero de 2008 a Junio de 2011. Investigdor principal: Dr. José M. Puerta.
10/2010–?	Miembro del Grupo de Investigación en Sistemas Inteligentes y Minería de Datos (SIMD). Instituto de Investigación en Informática de Albacete (I3A).

Cursos específicos recibidos

06/2008	*ECLIPSE en el soporte al desarrollo de software profesional.* Organizado por la Escuela Superior de Ingeniería Informática de la UCLM. 20 horas (2 créditos)
07/2009	*Robótica Móvil: una apuesta de futuro.* Organizado por el Vicerrectorado de Extensión Universitaria de la UCLM. 20 horas (2 créditos)

Idiomas

Español	Lengua materna
Inglés	lee/habla/escribe correctamente

Referencias

Las siguientes personas pueden informar sobre mis calificaciones profesionales, así como mi personalidad y capacidades de trabajo:

Prof. José Luis Sánchez García
Supervisor Beca Colaboración
Dpto. de S.I., UCLM
Campus Universitario s/n
Albacete, 02071, España
Tfno: +34 967599200 Ext. 2439
Fax: +34 967599224
Correo-e: jsanchez@dsi.uclm.es

Prof. Luis de la Ossa Jiménez
Supervisor Proyecto Fin de Carrera
Dpto. de S.I., UCLM
Campus Universitario s/n
Albacete, 02071, España
Tfno: +34 967599200 Ext. 2413
Fax: +34 967599224
Correo-e: ldelaossa@dsi.uclm.es

Albacete, 17 de Diciembre de 2010

Fdo. Javier Cózar del Olmo

Bibliografía

[1] J. Casillas, O. Cordón, I. Fernández de Viana, and F. Herrera. Learning cooperative linguistic fuzzy rules using the best-worst ant system algorithm. *International Journal of Intelligent Systems*, 20:433–452, 2005.

[2] J. Casillas, O. Cordón, and F. Herrera. COR: A methodology to improve ad hoc data-driven linguistic rule learning methods by inducing cooperation among rules. *IEEE Transactions on Systems, Man and Cybernetics. Part B: Cybernetics.*, 32(4):526–537, 2002.

[3] J. Casillas, O. Cordón, and F. Herrera. Different approaches to induce cooperation in fuzzy linguistic models under the COR methodology. pages 321–334, 2002.

[4] O. Cordón and F. Herrera. A proposal for improving the accuracy of linguistic modeling. *IEEE Transactions on Fuzzy Systems*, 8(6):335–344, 2000.

[5] L. delaOssa, J.A. Gámez, and J.M. Puerta. Learning cooperative fuzzy rules using fast local search algorithms. In *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2010)*, pages 2134–2141, 2006.

[6] L. delaOssa, J.A. Gámez, and J.M. Puerta. Learning weighted linguistic fuzzy rules by using specifically-tailored hybrid estimation of distribution algorithms. *International Journal of Approximate Reasoning*, 50(3):541–560, 2009.

[7] F.W. Glover and G.A. Kochenberger. *Handbook of Metaheuristics (International Series in Operations Research & Management Science)*. Springer, January 2003.

[8] G. Klir and B. Yuan. *Fuzzy Sets and Fuzzy Logic. Theory and Applications*. Prentice Hall PTR, 1995.

[9] Ebrahim H. Mamdani. Applications of fuzzy algorithm for control a simple dynamic plant. In *Proceedings of the IEEE 121(12)*, pages 1585–1588, 1974.

[10] E.H. Mamdani and S. Assilian. An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, 7:1–13, 1975.

[11] K. Nozaki, H. Ishibuchi, and H. Tanaka. A simple but powerful heuristic method for generating fuzzy rules from numerical data. *Fuzzy Sets and Systems*, 86:251–270, 1997.

[12] M. Sugeno and G.T. Kang. Structure identification of fuzzy models. *Fuzzy Sets and Systems*, 28(1):15–33, 1985.

[13] T. Takagi and M. Sugeno. Fuzzy identification of systems and its application to modelling and control. *IEEE Transactions on Systems, Man and Cybernetics*, 15(1):116–132, 1985.

[14] M. Verhoeven and E. Aarts. Parallel local search. *Journal of Heuristics*, 1(1):43–65, 1995.

[15] L.X. Wang and J.M. Mendel. Generating fuzzy rules by learning from examples. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(6):1414–1427, 1992.

[16] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[17] L.A. Zadeh. Outline of a new approach to the analysis of complex systems and decisionprocesses. *IEEE Transactions on Systems, Man, and Cybernetics*, 3(1):28–44, 1973.

[18] L.A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning. *Information Science*, 8:199–249, 1975.